8 - Basi di dati sematiche

ATTENZIONE

Questo capitolo per mancanza di tempo non è completo ma sufficiente a sostenere l'esame, per chi volesse approfondire ci sarebbe il libro di testo menzionato all'inizio, principalmente tutta la quarta parte del libro.

Le basi di dati semantiche sono collezioni di dati che possono essere arricchite con metadati che possono avere delle definizioni formali definite in vocabolari condivisi chiamati ontologie. Nascono dall'evoluzione dei linguaggi a oggetti e dei linguaggi logici e dall'evoluzione del Semantic Web.
L’obbiettivo è fornire una descrizione dei dati (annotazioni) con conformazione irregolare attraverso linguaggi con capacità espressiva crescente in grado di fornire informazioni dalle annotazioni. L’innovazione consiste nella costruzione di collezioni di dati aperte e connesse (linked open data).

Modello dei dati RDF

L'RDF È un modello per rappresentare informazioni sul Web sotto forma di **triple** (soggetto, predicato, oggetto), che formano un grafo orientato. Un esempio di tripla RDF è la frase:

$$< VerdiHaCompostoTraviata >$$

Le triple possono essere rappresentate graficamente mediante un grafo nel quale, per ogni tripla (s, p, o), esiste un arco orientato con etichetta p tra un nodo con etichetta s e un nodo con etichetta o. In genere, le etichette usate in RDF possono appartenere a un vocabolario o un’ontologia predefinita, e ciò può consentire di derivare ulteriore conoscenza. Ogni vocabolario (namespace) utilizzato in un’stanza di RDF è introdotto da un’istruzione prefix che consente di estrarre le etichette presenti nel vocabolario. Per esempio con il comando $@\mathrm{prefix\ o:} <\mathrm{http://www.polimi.it/ceri/opera}>$ si assegna al prefisso ‘o:’ l’indirizzo web dell’ontologia che si trova appunto tramite quel link.
La differenza con le basi di dati relazionali consiste nel fatto che, per queste ultime i dati possono essere rappresentati esclusivamente con uno schema corrispondente, mentre con le basi semantiche i dati hanno significato a prescindere dallo schema corrispondente, il quale può anche non essere definito.

I dati rappresentabili in RDF sono di tre categorie:

1. **IRI (Internationalized Resource Identifier)**: sono stringhe che identificano univocamente tutte le risorse disponibili sul Web in base a una notazione standard.
2. **Letterali**: sono valori che descrivono le risorse presenti nell’istanza RDF.

3. Blank nodes: sono identificatori usati per rappresentare risorse anonime, sintatticamente preceduti dal carattere underscore

Nelle triple RDF il soggetto può essere un IRI o un blank node, il predicato è un IRI e l'oggetto può essere IRI, letterale o blank node.
Esempio:

L'istanza precedente poteva essere scritta come sequenza di triple in conformità con la notazione RDF/XML, mentre con la notazione N3 quando due tuple condividono il soggetto si usa il separatore ';' (punto e virgola), quando condividono sia soggetto sia predicato, si usa ',' (virgola), ottenendo in questo modo una notazione più compatta

RDF Schema

Il modello di dati fin qui utilizzato è un modello utile a esprimere conoscenza elementare in forma di triple, ma non possiede di per se la capacità di classificare risorse sulla base delle loro caratteristiche semantiche. L'aggiunta della semantica è possibile con l'estensione RDFS di RDF, che aggiunge a RDF costrutti per descrivere **metadati**, ovvero struttura e proprietà di istanze RDF. Per ciò, RDFS costituisce lo strumento per definire lo schema di un'istanza RDF. I principali costrutti messi a disposizione da RDFS sono le **classi** e le **proprietà**. Una classe RDFS consente di definire insiemi di risorse di un'istanza RDF che hanno caratteristiche comuni. Una proprietà RDFS consente di definire il tipo di soggetto e oggetto dei predicati in un'istanza RDF (in termini matematici significa definire dominio e range di una funzione).

Esempio:
Il RDFS è inoltre possibile definire relazioni di generalizzazione tra classi e tra proprietà

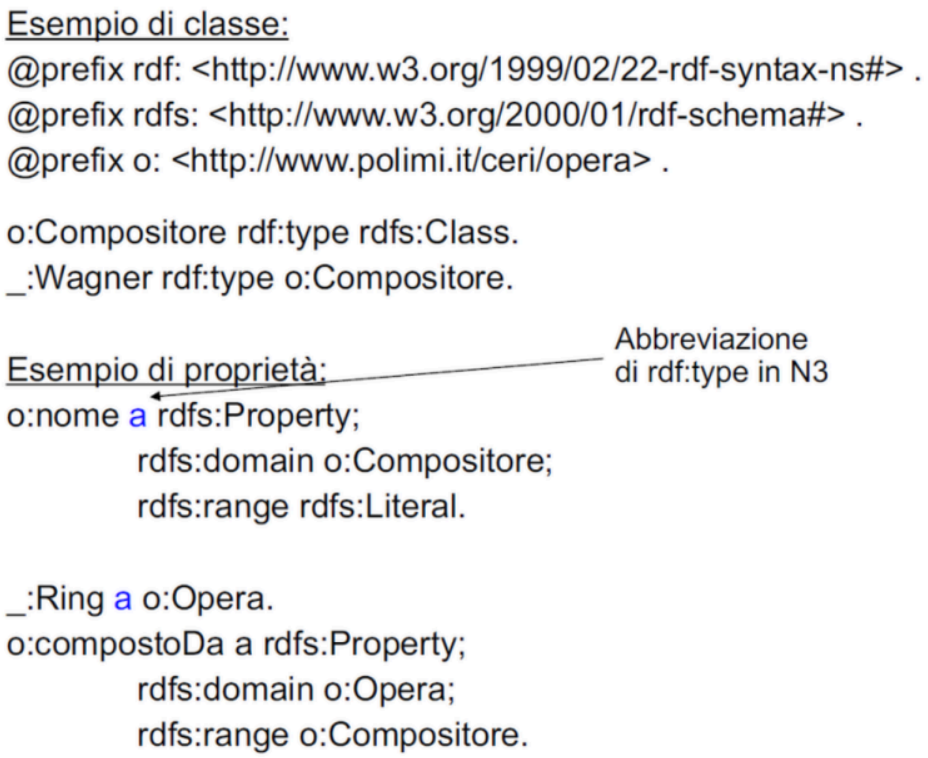

Figura 10.2: Esempio di istanza RDFS.

```
o:Artista rdf:type rdfs:Class.
o:Compositore rdfs:subClassOf o:Artista.
```

La semantica di RDFS definisce l'ereditarietà, infatti se i compositori sono artisti e gli artisti sono persone, allora i compositori sono persone. Similmente si può definire come casi particolari di altre proprietà utilizzando il predicato$\mathrm{rdfs:SubPropertyOf}$.

SPARQL 1.0

SPARQL (Simple Protocol and RDF Query Language) è un linguaggio di interrogazione con caratteristiche molto simili a SQL, proposto dal W3C per interrogare dati descritti in RDF. In questo linguaggio, alle classiche forme di interrogazione si affiancano altre forme:

- SELECT per interrogazioni classiche
- $\mathrm{DESCRIBE}$ per ottenere descrizione di risorse in un istanza RDF interrogabile in SPARQL (detto endpoint)
- ASK per sapere se specifici termini sono disponibili nell'endpoint
- $\mathrm{CONSTRUCT}$ per costruire un nuovo grafo RDF a partire da una interrogazione

Sintassi

Una query in SPARQL 1.0 è composta da cinque parti:

1. Clausola opzionale PREFIX per introdurre vocabolari

2. Il risultato prodotto dalla query
3. Gli endpoin consultati, tramite le clausole FROM e FROM NAMED, indicano una IRI da cui leggere l'istanza RDF
4. La parte centrale della query costituita dalla clausola WHERE, consente di esprimere condizioni di pattern matching tra la query stessa (triple pattern) e l'istanza RDF su cui opera
5. Modificatori opzionali, quali costrutti ORDER BY, LIMIT, OFFSET

La clausola FROM indica semplicemente una IRI da cui leggere i dati, la clausola from NAMED indica una URI in cui è presente un grafo RDF che viene associato a un nome. La parte centrale di un'interrogazione SPARQL, introdotta dalla clausola WHERE, consente di esprimere condizioni di pattern matching tra la query stessa e il grafo RDF su cui la query opera. Elemento centrale di un pattern matching è un **triple pattern**, che sintatticamente è una tripla nelle cui posizioni è possibile far comparire, in aggiunta a IRI, letterali e blank nodes, anche **variabili**. Per esempio un triple pattern è:

$$<?o1\ o{:}HaPersonaggio\ 'Sigrido'>$$

Vari triple pattern possono essere combinati assieme, formando un graph pattern.
I graph pattern possono essere estesi utilizzando alcune clausole opzionali:

- La clausola FILTER applica alle tuple di binding che risultano dalla valutazione della clausola WHERE un predicato; solo le tuple che soddisfano il predicato fanno parte del risultato finale.
  Gli operatori utilizzabili all'interno della clausola FILTER sono: Operatori di confronto, aritmetici, logici e predicati unari specifici di SPARQL
- La clausola UNION all'interno della clausola WHERE consente di eseguire l'unione dei risultati di due graph pattern.
- La clausola OPTIONAL consente di definire ulteriori graph pattern rispetto a quelli della clausola WHERE senza vincolare le variabili ivi introdotte ad assumere un valore preciso. Nel caso il valore manchi le variabili sono avvalorate con NULL

Oltre alla select, esistono altre forme di query. La CONSTRUCT consente di costruire un nuovo grafo RDF, con un meccanismo simile a quello delle view in SQL. La query ASK consente di valutare se la query ha un risultato non nullo, cioè se alle variabili della query viene associata almeno una tupla di binding. Infine, la query DESCRIBE consente di estrarre tutta l'informazione conosciuta relativamente alle risorse che soddisfano una query.

SPARQL 1.1

La definizione di SPARQL 1.0 difettava di alcune caratteristiche tipiche dei linguaggi di query (in particolare di SQL); tali aspetti sono stati recentemente introdotti in SPARQL 1.1. Nello specifico introduce:

- Le clausole GROUP BY e HAVING

- L’operatore binario MINUS per effettuare la differenza
- La clausola NOT EXISTS che ricorda la sottoquery di SQL
- Le nozioni di entailment e property path che estendono in modo significativo il potere espressivo
- L’operatore SERVICE che consente forme sofisticate di interoperabilità

Focalizziamo l’attenzione sull'estensione **dell'entailment regimes** che è l’innovazione principale di SPARQL 1.1. Gli entailment regime sono particolari contesti di valutazione che estendono la valutazione delle query includendo nel risultato tutte le triple che sono implicate dalla istanza in base agli entailments RDFS. Fino ad’ora con SPARQL 1.0 abbiamo specificato un graph pattern, applicato un pattern matching tra il graph patter e la collezione di dati RDF ed il risultato viene restituito solo nel caso in cui ci fosse una corrispondenza 1 a 1 tra il graph pattern specificato e la base di dati.
Con l’entailment regime non viene restituito solo quello che trovo come conoscenza asserita, ma viene restituito anche quello che posso derivare (o inferire). Si cerca, quindi, non solo la conoscenza asserita, ma anche tutta la conoscenza che si può ottenere dall'applicazione di un reasoner deduttivo.
Supponendo che lo schema RDF comprenda un predicato amico, che collega fra loro blank nodes corrispondenti a persone, e supponendo che tali blank nodes abbiano una relazione HaNome, l’interrogazione percorre transitivamente la relazione a partire dalla persona di nome Giorgio.

Open Data

I Linked Data sono risorse disponibili sul Web, descritte tramite triple RDF e collegate fra loro tramite riferimenti (link). Pubblicare risorse sotto forma di Linked Data vuol dire aderire ad una buona pratica di pubblicazione, usando le IRI per identificare le risorse, lo standard HTTP per trasferire dati tra diversi nodi del Web, e gli standard RDF, RDFS e OWL per descrivere le triple.
Differenze tra open data e linked open data:

- I linked open data sono dati in formato aperto interconnessi tra loro.
- Gli open data sono semplicemente dati in formato aperto non necessariamente connessi tra di loro
- Con la prima categoria, i dati oltre ad essere in formato aperto fanno uso di linguaggi standard che assicurano la portabilità (RDF e RDFS), sono interconnessi tra loro utilizzando vocabolari già esistenti e tramite l’utilizzo di IRI.