RDFS

- RDF Schema = Lenguaje de descripción de vocabularios RDF
- RDFS no proporciona vocabularios específicos, sino facilidades para describir las clases y propiedades de un dominio específico.
- Las facilidades que proporciona RDFS se expresan como un vocabulario RDF, es decir, un conjunto especializado de recursos RDF con sus propios significados.
- RDFS puede verse como un *sistema de tipos* para RDF, similar al de los lenguajes orientados a objetos: recursos como ejemplares de una o más clases, jerarquía de clases, propiedades asociadas a clases...

Vocabulario para declarar clases: `rdfs:Class`

Clase: cualquier recurso que tiene una propiedad `rdf:type` cuyo valor es el recurso `rdfs:Class`

- En N3:

```
:Libro   a   rdfs:Class .   (o bien
:Libro   rdf:type   rdfs:Class .)
```

- En XML:

```
<rdf:Description  rdf:ID="Libro">
  <rdf:type
   rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
</rdf:Description>
```

O bien, abreviado:

```
<rdfs:Class rdf:ID="Libro" />
```

Vocabulario para construir taxonomías: `rdfs:subClassOf`

Taxonomía: Clasificación de recursos en forma de jerarquía de clases.

Ontología: (en RDFS) Taxonomía +

- propiedades de las clases +
- a qué clases puede aplicarse cada propiedad (dominio) +
- qué valores puede tener cada propiedad (rango)

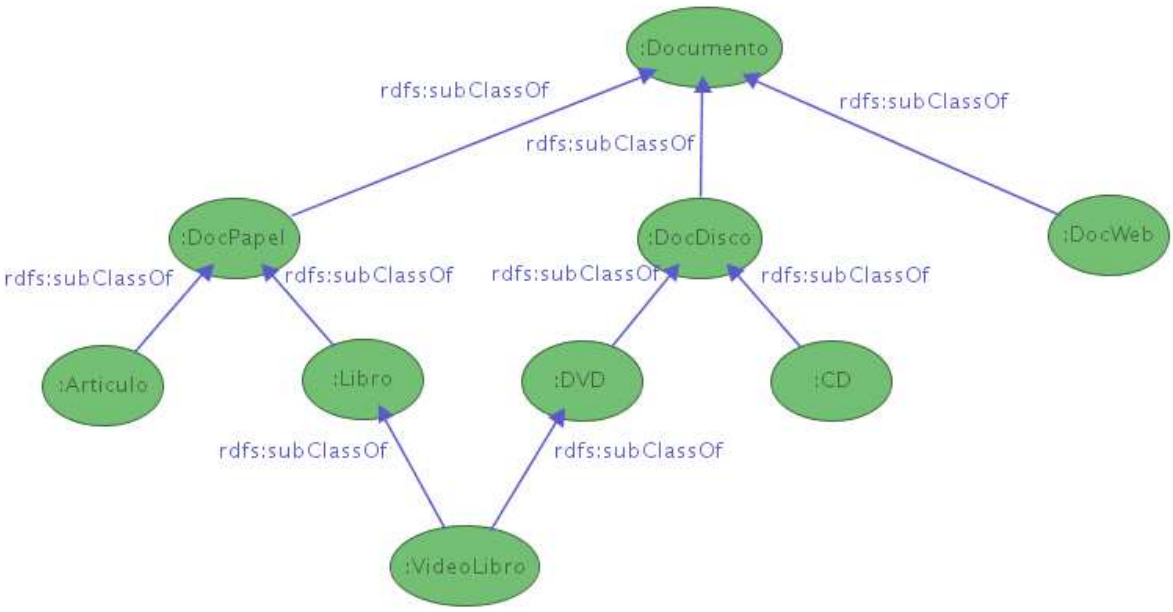
- Con `rdfs:subClassOf` se pueden definir jerarquías de clases

  `:LibroTecnico rdfs:subClassOf :Libro .`

- Una clase puede ser subclase de varias clases

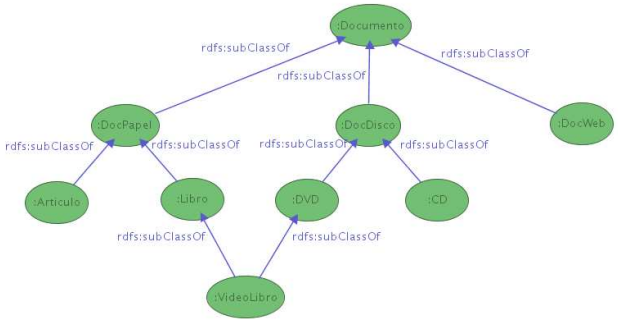
  `:LibroTecnico rdfs:subClassOf :DocsTecnicos .`

Ejemplo de taxonomía



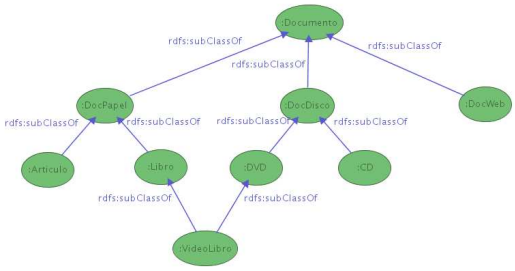
Ejemplo de taxonomía: RDF/N3

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix : <http://www.example.org/docs#> .
:Documento      a     rdfs:Class .
:DocPapel       a     rdfs:Class ;
   rdfs:subClassOf    :Documento .
:DocDisco       a     rdfs:Class ;
   rdfs:subClassOf    :Documento .
:DocWeb         a     rdfs:Class ;
   rdfs:subClassOf    :Documento .
:Libro          a     rdfs:Class ;
   rdfs:subClassOf    :DocPapel .
:Articulo       a     rdfs:Class ;
   rdfs:subClassOf    :DocPapel .
:CD             a     rdfs:Class ;
   rdfs:subClassOf    :DocDisco .
:DVD            a     rdfs:Class ;
   rdfs:subClassOf    :DocDisco .
:VideoLibro     a     rdfs:Class ;
   rdfs:subClassOf    :DVD ;
   rdfs:subClassOf    :Libro .
```



Ejemplo de taxonomía: RDF/XML

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xml:base="http://www.example.org/docs" >
  <rdfs:Class rdf:ID="Documento">
  </rdfs:Class>
  <rdfs:Class rdf:ID="DocPapel">
    <rdfs:subClassOf rdf:resource="#Documento" />
  </rdfs:Class>
  <rdfs:Class rdf:ID="DocWeb">
    <rdfs:subClassOf rdf:resource="#Documento" />
  </rdfs:Class>
  <rdfs:Class rdf:ID="DocDisco">
    <rdfs:subClassOf rdf:resource="#Documento" />
  </rdfs:Class>
  <rdfs:Class rdf:ID="Libro">
    <rdfs:subClassOf rdf:resource="#DocPapel" />
  </rdfs:Class>
  <rdfs:Class rdf:ID="Articulo">
    <rdfs:subClassOf rdf:resource="#DocPapel" />
  </rdfs:Class>
  <rdfs:Class rdf:ID="CD">
    <rdfs:subClassOf rdf:resource="#DocDisco" />
  </rdfs:Class>
  <rdfs:Class rdf:ID="DVD">
    <rdfs:subClassOf rdf:resource="#DocDisco" />
  </rdfs:Class>
  <rdfs:Class rdf:ID="VideoLibro">
    <rdfs:subClassOf rdf:resource="#Libro" />
    <rdfs:subClassOf rdf:resource="#DVD" />
  </rdfs:Class>
</rdf:RDF>
```



Clases y ejemplares (o *instances*, o miembros, o individuos)

Si usando el vocabulario anterior declaramos:

```
<docs:VideoLibro ID="unVideoLibro">
 ...
</docs:VideoLibro>
```

Es decir:

```
:unVideoLibro   a   docs:VideoLibro ;
                ...   .
```

Un agente puede inferir:

```
:unVideoLibro   a   docs:Libro ;
                a   docs:DVD .
```

Y, como `rdfs:subClassOf` es transitiva,

```
:unVideoLibro   a   docs:DocPapel ;
                a   docs:DocDisco ;
                a   docs:Documento .
```


Definición y uso de clases

- Definición en el «esquema» de *Novela* como tipo de datos de la aplicación (en DL, «axioma terminológico»: *Novela* $\sqsubseteq$ *Libro*):

```
<rdfs:Class rdf:ID="Novela">
   <rdfs:subClassOf rdf:resource="#Libro" />
</rdfs:Class>
```

- Descripción de una novela concreta, ejemplar de `Novela` (en DL, una «aserción»: *Novela(urn:isbn:1234567890)*).
  Si el anterior fragmento forma parte de un esquema localizable en `http://www.example.org/docs`,

```
<rdf:Description rdf:about="urn:isbn:1234567890" >
  <rdf:type rdf:resource="http://www.example.org/docs#Novela" />
   ...
</rdf:Description>
```

  O, en forma abreviada:

```
<docs:Novela rdf:about="urn:isbn:1234567890" >
  ...
</docs:Novela>
```

Píldora lógica: subsunción e inferencias

RDFS/N3: `ns1:Clase1 rdfs:subClassOf ns2:Clase2`
FOL: $(\forall x)(c_1(x) \Rightarrow c_2(x))$
DL: Clase1$\sqsubseteq$ Clase2 («la Clase2 *subsume* a la Clase1»)

Reglas de inferencia:

- En FOL, *modus ponens*:
  Premisa1: $(\forall x)(c_1(x) \Rightarrow c_2(x))$ (una «regla»)
  Premisa2: $c_1(A)$ (un «hecho»)
  Conclusión: $c_2(A)$
- En DL:
  Premisa1: Clase1$\sqsubseteq$ Clase2 (un «axioma»)
  Premisa2: Clase1(A) (una «aserción»)
  Conclusión: Clase2(A)
- En RDF/RDFS:
  Un agente tiene acceso a una ontología que contiene
  `ns1:Clase1 rdfs:subClassOf ns2:Clase2 .`
  El agente recibe el dato: `<URIrecurso> a ns1:Clase1 .`
  El agente infiere la tripla `<URIrecurso> a ns2:Clase2 .`

Raíz de todas las clases: `rdfs:Resource`

Todas las clases son subclases de la clase `rdfs:Resource`

- `rdfs:Resource` es la clase raíz de todas las clases, y, a su vez, es un miembro de `rdfs:Class`.
- `rdfs:Class` es la clase de recursos que son clases. *Es un miembro de sí misma.*
  $\rightsquigarrow$ Paradoja de Russell.

Nota:

$$\texttt{rdfs:Resource} \neq \texttt{rdf:resource}$$

`rdf:resource` es un atributo XML cuyo valor identifica a un recurso particular (el objeto de una tripla).

Píldora lógica: paradoja de Russell, o paradoja del barbero (1902)

`rdfs:Resource` es lo que en DL se llama «$\top$»
I(`rdfs:Resource`) = I($\top$) = $\mathcal{U}$
I(`rdfs:Class`) = $\mathcal{P}(\mathcal{U})$

- «Todas las clases son subclases de la clase `rdfs:Resource`»: $C \subseteq \mathcal{U}$
- «`rdfs:Resource` es un miembro de `rdfs:Class`»: $\mathcal{U} \in \mathcal{P}(\mathcal{U})$
- «`rdfs:Class` es la clase de recursos que son clases. Es un miembro de sí misma»: $\mathcal{P}(\mathcal{U}) \in \mathcal{P}(\mathcal{U})$

- Conjuntos normales: los que no se contienen a sí mismos: $N = \{x | x \notin x\}$ (1)
- Conjuntos singulares: los que se contienen a sí mismos

El conjunto N ¿se contiene a sí mismo? Si no se contiene a sí mismo, pertenece al tipo de conjuntos N y por lo tanto se contiene a sí mismo. Es decir, se contiene a sí mismo sólo si no se contiene a sí mismo.

Según la teoría de conjuntos de Cantor (1) se puede expresar así: $(\forall x)(x \in N \Leftrightarrow x \notin x)$
Y como N es un conjunto, particularizando $x = N$ resulta: $N \in N \Leftrightarrow N \notin N)$
$\rightsquigarrow$ Inconsistencia de la teoría de Cantor y Frege, resuelta con la teoría de los tipos de Russell.

Propiedades

Todas las propiedades son ejemplares (miembros) de la clase `rdf:Property`

```
:numPag  a        rdf:Property .
:numPag  rdf:type rdf:Property .
```

(o bien:)

En XML:

```
<rdf:Description  rdf:ID="numPag">
 <rdf:type
 rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
 ...
</rdf:Description>
```

O bien:

```
<rdf:Property rdf:ID="numPag" />
...
</rdf:Property>
```

Píldora lógica: semántica de las propiedades

`rdf:Property` es una clase:

I(`rdf:Property`) = $\mathcal{P} \subseteq \mathcal{U}$

«Todas las propiedades son miembros de `rdf:Property`»:

I(p) = $p \in \mathcal{P}$

Luego $p \in \mathcal{U}$, cuando debería ser:

$p \in \mathcal{U} \times \mathcal{U}$ (o $p \in \mathcal{U} \times \mathcal{D}$)

Vocabulario para asociar propiedades a clases

- Para indicar que una propiedad se aplica a una determinada clase se usa la *propiedad RDFS* `rdfs:domain`, y para indicar que sus valores son ejemplares de otra clase se usa la *propiedad RDFS* `rdfs:range`:

```
:Documento  a            rdfs:Class .
:Persona    a            rdfs:Class .
:autor      a            rdf:Property ;
            rdfs:domain  :Documento ;
            rdfs:range   :Persona .
```

- `rdfs:range` se usa también para indicar que el valor es un literal con tipo:

```
:numPag     a            rdf:Property ;
            rdfs:range   xsd:nonNegativeInteger .
```

`rdfs:domain` y `rdfs:range` son propiedades...

Declaraciones implícitas en RDFS:

```
rdfs:domain     rdfs:domain    rdf:Property ;
                rdfs:range     rdfs:Class .
rdfs:range      rdfs:domain    rdf:Property ;
                rdfs:range     rdfs:Class .
```

Es decir,

- se aplican a recursos que son propiedades
  (todo recurso que tenga una propiedad
  `rdfs:domain` o `rdfs:range` es un ejemplar de `rdf:Property`)
- sus valores son clases
  (todo recurso que sea el valor de una propiedad
  `rdfs:domain` o `rdfs:range` es un ejemplar de `rdfs:Class`).

Subpropiedades

«`P1 rdfs:subPropertyOf P2`» significa que todas la parejas de recursos relacionadas por P1 están también relacionadas por P2:

```
:autor          a                    rdf:Property .
:primerAutor    a                    rdf:Property ;
                rdfs:subPropertyOf   :autor .
```

Una propiedad (o subpropiedad) puede tener ninguna, una o varias propiedades de rango:

```
:autora     a                    rdf:Property ;
            rdfs:subPropertyOf   :autor ;
            rdfs:range           :Mujer ;
            rdfs:range           :Persona .
```

Cuando se usa esta propiedad, por ejemplo, en:

```
:doc1 docs:autora :aa1
```

para que la sentencia sea correcta, `:aa1` debe ser un ejemplar tanto de `docs:Mujer` como de `docs:Persona`

Igualmente, una propiedad puede tener ninguna, una o varias propiedades de dominio.

¿Qué es una clase?

- **En OOP:** «Una clase define las características abstractas de una cosa (objeto): atributos (o propiedades, o campos) y comportamiento (métodos)» (Wikipedia).
- **En RDFS:** «Una clase es cualquier recurso que tiene una propiedad `rdf:type` cuyo valor es el recurso `rdfs:Class`» (RDF Primer: `http://www.w3.org/TR/rdf-primer/`).

Diferencias:

- EN RDFS no hay comportamiento asociado a los objetos (recursos), y, por tanto, a las clases (es un modelo de datos).
- En OOP son los atributos y el comportamiento los que definen una clase. En RDFS las clases se declaran independientemente de los atributos (propiedades).

La segunda diferencia está relacionada con la perspectiva de *mundo abierto* de los modelos RDFS.

RDFS vs. OO (1)

Diferencia de enfoque:

- *En los lenguajes orientados a objetos se definen las clases en función de las propiedades que pueden tener sus ejemplares*. Ejemplo: definición de la clase «Documento» con un atributo «autor» del tipo «Persona».
- *En RDFS se definen las propiedades en función de las clases de recursos a las que se aplican*. Ejemplo: si se ha definido «autor» con un dominio «Documento» y un rango «Persona», «Los autores de d1 y d2 son a1 y a2»:

```
:d1  docs:autor  :a1 ;
     docs:autor  :a2 .
:d2  docs:autor  :a1 ;
     docs:autor  :a2 .
```

RDFS vs. OO (2)

Ventaja del «enfoque centrado en propiedades» de RDFS: permite extender fácilmente la descripción de recursos existentes. Ej: integración de otras propiedades («editor», «precio», «traductor»...) con el mismo dominio sin necesidad de redefinir la descripción original de la clase.

«Los autores de l1 y l2 son a1 y a2»

```
:d1  docs:autor  :a1 ;    :d2  docs:autor  :a1 ;
     docs:autor  :a2 .         docs:autor  :a2 .
```

De otra fuente: «El precio de d1 es p1, y el de d2 es p2»

```
:d1  docs:autor    :a1 ;   :d2  docs:autor    :a1 ;
     docs:autor    :a2 ;        docs:autor    :a2;
     libros:precio :p1 .        libros:precio :p2 .
```


Definición de propiedades

Una propiedad se define, normalmente, con tres triplas.

En N3:

```
 :autor    a             rdf:Property ;
           rdfs:domain   :Documento ;
           rdfs:range    :Persona .
```

En XML:

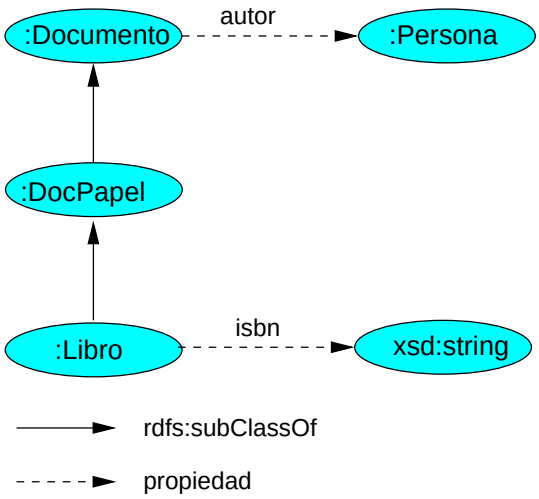
```
<rdf:Description  rdf:ID="autor">
 <rdf:type  rdf:resource=
            "http://www.w3.org/1999/02/22-rdf-syntax-ns#Property" />
 <rdfs:domain  rdf:resource="#Documento" />
 <rdfs:range  rdf:resource="#Persona" />
</rdf:Description>
```

En XML, forma abreviada:

```
<rdf:Property rdf:ID="autor" />
  <rdfs:domain  rdf:resource="#Documento" />
  <rdfs:range  rdf:resource="#Persona" />
</rdf:Property>
```

Grafos simplificados

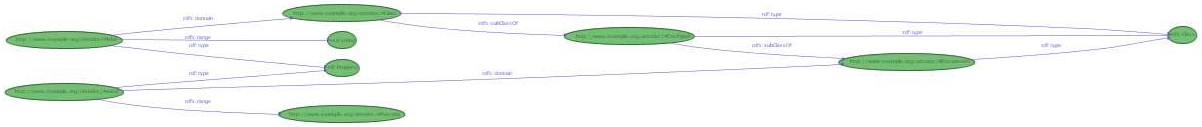
```
@prefix : <http://www.example.org/docs#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
 :Documento  a   rdfs:Class .
 :DocPapel   a   rdfs:Class;
             rdfs:subClassOf :Documento .
 :Libro  a rdfs:Class;
         rdfs:subClassOf :DocPapel .

 :autor  a  rdf:Property;
         rdfs:domain :Documento;
         rdfs:range  :Persona .

 :isbn   a  rdf:Property;
         rdfs:domain :Libro;
         rdfs:range  xsd:string .
```



Útil, pero *no es un grafo RDF*

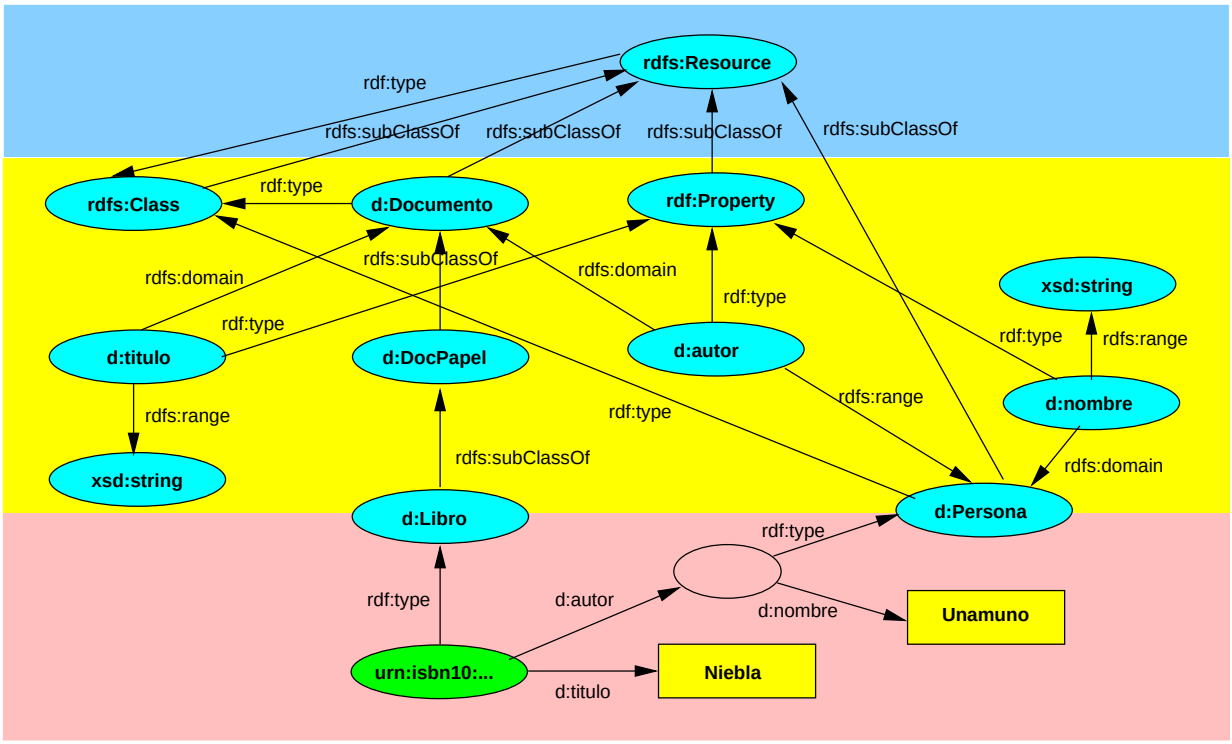
Grafos RDF

Grafo y RDF generados por IsaViz a partir del N3 anterior:



```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  <rdfs:Class rdf:about="http://www.example.org/docs#Libro">
    <rdfs:subClassOf>
      <rdfs:Class rdf:about="http://www.example.org/docs#DocPapel"/>
    </rdfs:subClassOf>
  </rdfs:Class>
  <rdfs:Class rdf:about="http://www.example.org/docs#Documento"/>
  <rdfs:Class rdf:about="http://www.example.org/docs#DocPapel">
    <rdfs:subClassOf rdf:resource="http://www.example.org/docs#Documento"/>
  </rdfs:Class>
  <rdf:Property rdf:about="http://www.example.org/docs#autor">
    <rdfs:domain rdf:resource="http://www.example.org/docs#Documento"/>
    <rdfs:range rdf:resource="http://www.example.org/docs#Persona"/>
  </rdf:Property>
  <rdf:Property rdf:about="http://www.example.org/docs#isbn">
    <rdfs:domain rdf:resource="http://www.example.org/docs#Libro"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </rdf:Property>
</rdf:RDF>
```

Grafo de un esquema y de su uso



rdfs:Literal

Clase cuyos ejemplares no se identifican por un URI, sino por una representación léxica.

- Literales simples *(plain literals)*: una cadena opcionalmente acompañada de una etiqueta de lenguaje.
- Literales con tipo *(typed literals)*: una cadena necesariamente acompañada de un URI que identifica el tipo de datos.
  Un literal con tipo denota al miembro del tipo de datos que resulta de aplicar a la cadena la función de correspondencia entre la representación léxica y el valor *(lexical-to-value mapping)*.

Un literal con tipo es un miembro de alguna clase de tipo de datos (`xsd:boolean`, `xsd:date`, `xsd:integer`, `xsd:float`...)

`http://www.w3.org/TR/2004/REC-rdf-mt-20040210/#DTYPEINTERP`

rdfs:Datatype

Clase cuyos ejemplares son los distintos tipos de datos. Cada uno de estos ejemplares es una subclase de `rdfs:Literal`

Para especificar que un recurso es un ejemplar de `rdfs:Datatype`:

En N3:

```
:numPag        a  rdf:Property;
               rdfs:domain :DocPapel;
               rdfs:range  xsd:integer .
xsd:integer    a  rdfs:Datatype .
```

En RDF/XML: `<rdfs:Datatype  rdf:about=&xsd;integer" />`

En los documentos que usen este vocabulario se debe especificar el tipo:

En N3: `<urn:isbn:0486600289>  docs:numPag  217^^xsd:integer ;`

En RDF/XML:

```
<rdf:Description about="urn:isbn:0486600289">
   <docs:isbn rdf:datatype="&xsd;integer">217</docs:isbn>
    ...
</rdf:Description>
```


rdfs:seeAlso y rdfs:isDefinedBy

Propiedades (es decir, ejemplares de `rdf:Property`) predefinidas en RDFS.

Ambas tienen como dominio y rango `rdfs:Resource` y se usan para apuntar a descripciones alternativas del sujeto:

- El objeto de `rdfs:seeAlso` es otro recurso que puede dar información adicional sobre el sujeto.
- `rdfs:isDefinedBy` es una subpropiedad de `rdfs:seeAlso` más específica: el objeto es la fuente original de la información

Son a RDF lo que el elemento «anchor» es a HTML

`rdfs:label` y `rdfs:comment`

Como las anteriores, son *propiedades* predefinidas en RDFS.

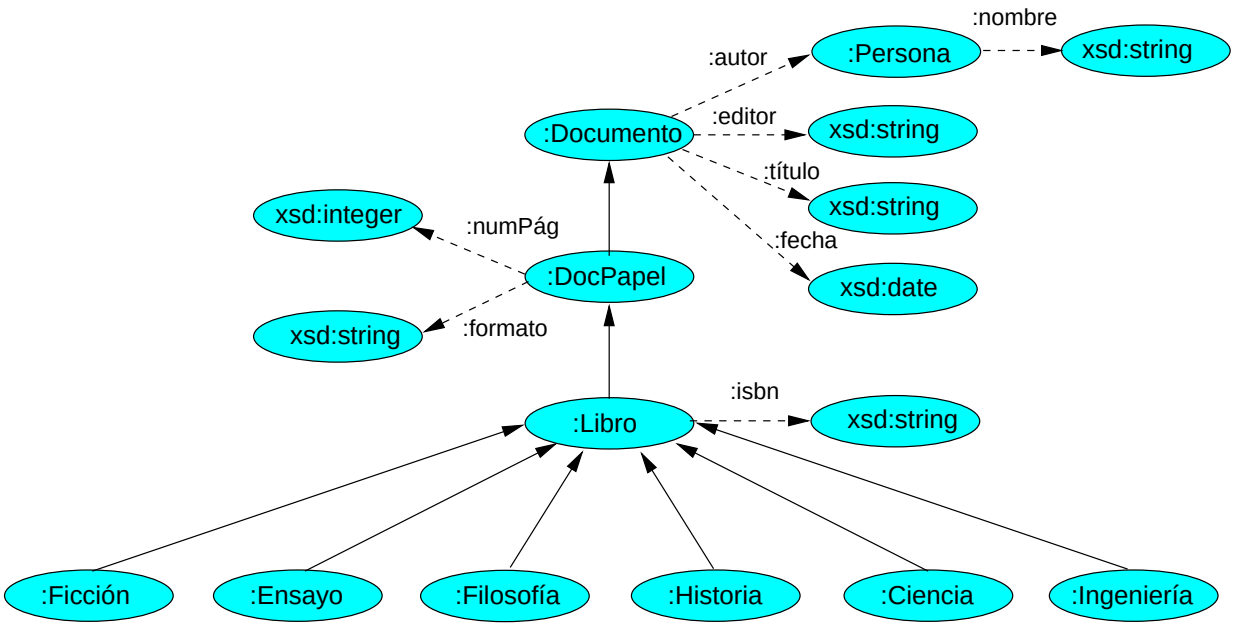
Ambas tienen:

- `rdfs:domain` `rdfs:Resource`
- `rdfs:range` `rdfs:Literal`

Y se usan para:

- `rdfs:label`: dar una *versión* humanamente inteligible del *nombre* de un recurso
- `rdfs:comment`: dar una *descripción* humanamente inteligible de un recurso

Ontolib1



¿Clases o propiedades?

«Ficción», «Ensayo»... Tiene sentido que sean clases si vamos a definir propiedades específicas para ellas.

Hay varias reglas prácticas que ayudan a decidir cuándo introducir clases nuevas en una jerarquía. Normalmente, las subclases de una clase

1. *tienen propiedades adicionales que la superclase no tiene, o*
2. *tienen restricciones diferentes de las de la superclase, o*
3. *participan en relaciones diferentes de las de la superclase*

Sin embargo, a veces resulta útil crear clases aunque no tengan nuevas propiedades: las clases en las jerarquías terminológicas no tienen por qué introducir nuevas propiedades.

«Ontology Development 101»

`http://ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness-abstract.html`

Ontolib1 en RDF/N3: clases

```
@prefix : <http://www.gsi.dit.upm.es/~gfer/ontologias/Ontolib1.rdfs#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
    :Persona     a rdfs:Class;
         rdfs:label "Persona";
         rdfs:comment "Mejor utilizar foaf:Person..."@es-ES .
    :Documento   a rdfs:Class .
    :DocPapel    a rdfs:Class;
         rdfs:subClassOf :Documento .
    :Libro       a rdfs:Class;
         rdfs:subClassOf :DocPapel .
    :Ficcion     a rdfs:Class;
         rdfs:subClassOf :Libro .
    :Ensayo     a rdfs:Class;
         rdfs:subClassOf :Libro .
    :Filosofia    a rdfs:Class;
         rdfs:subClassOf :Libro .
    :Historia    a rdfs:Class;
         rdfs:subClassOf :Libro .
    :Ciencia     a rdfs:Class;
         rdfs:subClassOf :Libro .
    :Ingenieria  a rdfs:Class;
         rdfs:subClassOf :Libro .
```

Ontolib1 en RDF/N3: propiedades

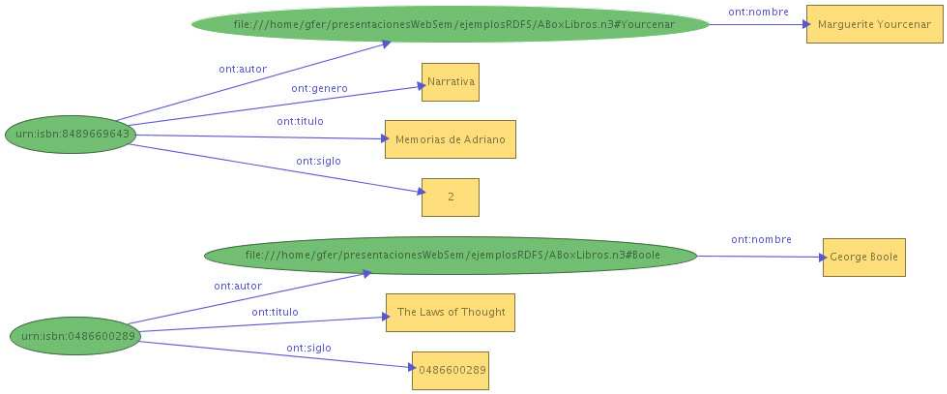
```
:nombre   a rdf:Property;
     rdfs:domain :Persona;
     rdfs:range xsd:string .
:autor    a rdf:Property;
     rdfs:domain :Documento;
     rdfs:range :Persona;
     rdfs:comment """
          "Mejor seria definir una coleccion o lista de autores
           mediante rdf:parseType='Collection'"
         """@es-ES .
:editor   a rdf:Property;
     rdfs:domain :Documento;
     rdfs:range xsd:string .
:titulo     a rdf:Property;
     rdfs:domain :Documento;
     rdfs:range xsd:string .
:fecha      a rdf:Property;
     rdfs:domain :Documento;
     rdfs:range xsd:date .
:numPag   a rdf:Property;
     rdfs:domain :DocPapel;
     rdfs:range xsd:integer .
:formato  a rdf:Property;
     rdfs:domain :DocPapel;
     rdfs:range xsd:string .
:isbn     a rdf:Property;
     rdfs:domain :Libro;
     rdfs:range xsd:string .
```


Ontolib1 en RDF/N3: otras propiedades

```
:genero   a rdf:Property;
     rdfs:domain :Ficcion;
     rdfs:range xsd:string .
:siglo    a rdf:Property;
     rdfs:domain :Historia;
     rdfs:range xsd:integer .
:tipoEns a rdf:Property;
     rdfs:domain :Ensayo;
     rdfs:range xsd:string .
:ramaFil a rdf:Property;
     rdfs:domain :Filosofia;
     rdfs:range xsd:string .
:areaCien a rdf:Property;
     rdfs:domain :Ciencia;
     rdfs:range xsd:string .
:areaIng a rdf:Property;
     rdfs:domain :Ingenieria;
     rdfs:range xsd:string .
xsd:integer a  rdfs:Datatype .
xsd:string  a  rdfs:Datatype .
xsd:date    a  rdfs:Datatype .
```

Ontolib1: ejemplares *(instances, ABox)*

```
@prefix ont: <http://www.gsi.dit.upm.es/~gfer/ontologias/Ontolib1.rdfs#> .
@prefix : <#> .
<urn:isbn:0486600289> ont:isbn   "0486600289"^^xsd:string ;
                      ont:titulo "The Laws of Thought"^^xsd:string ;
                      ont:autor  :Boole .
<urn:isbn:8489669643> ont:titulo "Memorias de Adriano"^^xsd:string ;
                      ont:autor  :Yourcenar;
                      ont:siglo  "2"^^xsd:integer;
                      ont:genero "Narrativa"^^xsd:string .
:Boole       ont:nombre  "George Boole"^^xsd:string .
:Yourcenar   ont:nombre  "Marguerite Yourcenar"^^xsd:string .
```



Ontolib1: inferencias

```
@prefix ont: <http://www.gsi.dit.upm.es/~gfer/ontologias/Ontolib1.rdfs#> .
<urn:isbn:0486600289> ont:isbn   "0486600289"^^xsd:string;
                      ont:autor  :Boole .
```

El agente puede inferir que:

- `<urn:isbn:0486600289>` es un libro, porque `Ontolib1` dice que
  `ont:isbn rdfs:domain ont:Libro.`
- `:Boole` es una persona, porque `ont:autor rdfs:range ont:Persona.`

```
<urn:isbn:8489669643> ont:autor  :Yourcenar;
                      ont:genero "novela"^^xsd:string;
                      ont:siglo  "2"^^xsd:integer.
```

El agente puede inferir que:

- `:Yourcenar` es una persona.
- `<urn:isbn:8489669643>` es un libro de ficción y de historia, porque
  `ont:genero rdfs:domain ont:Ficcion` y
  `ont:siglo rdfs:domain ont:Historia.`

Píldora lógica: inferencias por restricciones de rango

RDFS/N3: `:P rdfs:range :R`
FOL: $(\forall x_1, x_2)(p(x_1, x_2) \Rightarrow r(x_2))$
DL: $\top \sqsubseteq \forall P.R$

Reglas de inferencia:

- En FOL: Premisa1: $(\forall x_1, x_2)(p(x_1, x_2) \Rightarrow r(x_2))$ (una «regla»)
  Premisa2: $p(A, B)$ (un «hecho»)
  Conclusión: $r(B)$
- En DL: Premisa1: $\top \sqsubseteq \forall P.R$ (un «axioma»)
  Premisa2: $P(A, B)$ (una «aserción»)
  Conclusión: $R(B)$
- En RDF/RDFS:
  Un agente tiene acceso a una ontología (TBox) que contiene
  `ont:prop rdfs:range ont1:ClaseR .`
  El agente recibe un dato (ABox) en forma de tripla:
  `<URIsujeto> ont:prop <URIobjeto> .`
  El agente infiere la tripla `<URIobjeto> a ont1:ClaseR .`

Píldora lógica: inferencias por restricciones de dominio

RDFS/N3: `:P rdfs:domain :D`
FOL: $(\forall x_1, x_2)(p(x_1, x_2) \Rightarrow d(x_1))$
DL: $\exists P.\top \sqsubseteq D$

Reglas de inferencia:

- En FOL: Premisa1: $(\forall x_1, x_2)(p(x_1, x_2) \Rightarrow d(x_1))$ (una «regla»)
  Premisa2: $p(A, B)$ (un «hecho»)
  Conclusión: $d(A)$
- En DL: Premisa1: $\exists P.\top \sqsubseteq D$ (un «axioma»)
  Premisa2: $P(A, B)$ (una «aserción»)
  Conclusión: $D(A)$
- En RDF/RDFS:
  Un agente tiene acceso a una ontología (TBox) que contiene
  `ont:prop rdfs:domain ont1:ClaseD .`
  El agente recibe un dato (ABox) en forma de tripla:
  `<URIsujeto> ont:prop <URIobjeto> .`
  El agente infiere la tripla `<URIsujeto> a ont1:ClaseD .`

Reglas de inferencia en N3

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix log: <http://www.w3.org/2000/10/swap/log#> .

#Reglas de subsuncion:
{?A rdfs:subClassOf ?B. ?S a ?A} log:implies {?S a ?B}.
{?B rdfs:subClassOf ?C. ?A rdfs:subClassOf ?B}
                                    log:implies {?A rdfs:subClassOf ?C}.
{?P rdfs:subPropertyOf ?R. ?S ?P ?O} log:implies {?S ?R ?O}.
{?Q rdfs:subPropertyOf ?R. ?P rdfs:subPropertyOf ?Q}
                                    log:implies {?P rdfs:subPropertyOf ?R}.

#Reglas de dominio y rango:
{?P rdfs:domain ?C. ?S ?P ?O} log:implies {?S a ?C}.
{?P rdfs:range ?C. ?S ?P ?O} log:implies {?O a ?C}.

#Otras reglas:
{?S ?P ?O} log:implies {?S a rdfs:Resource}.
{?S ?P ?O} log:implies {?O a rdfs:Resource}.
{?S ?P ?O} log:implies {?P a rdf:Property}.
{?C a rdfs:Class} log:implies {?C rdfs:subClassOf rdfs:Resource}.
```


Inferencias en la web

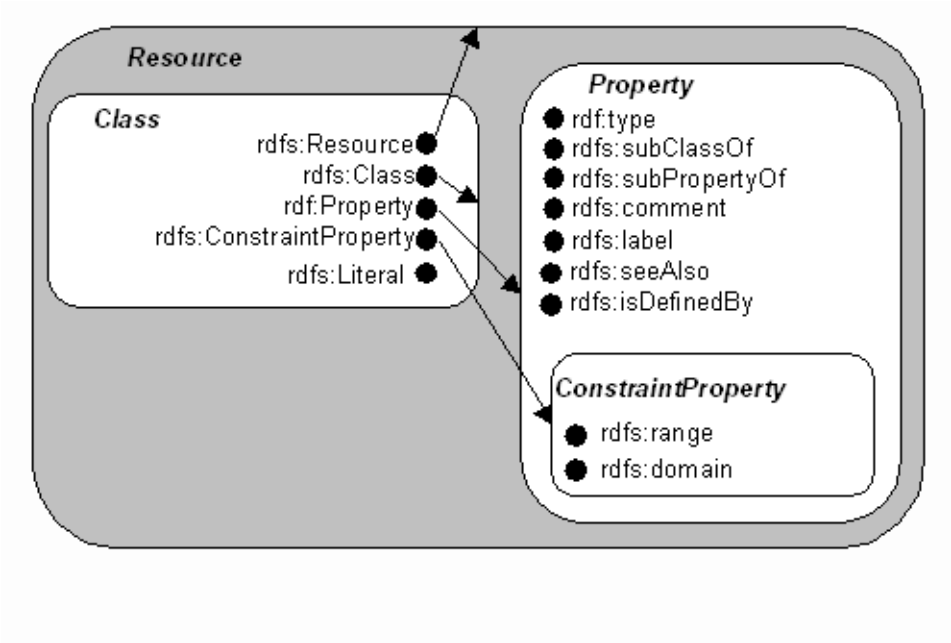
```
cwm --mode=r
    --rdf http://www.gsi.dit.upm.es/~gfer/ontologias/Ontolib1.rdfs
    --n3 ABoxLibros.n3  --think=reglas.n3
```

Resultado (entre otras cosas):

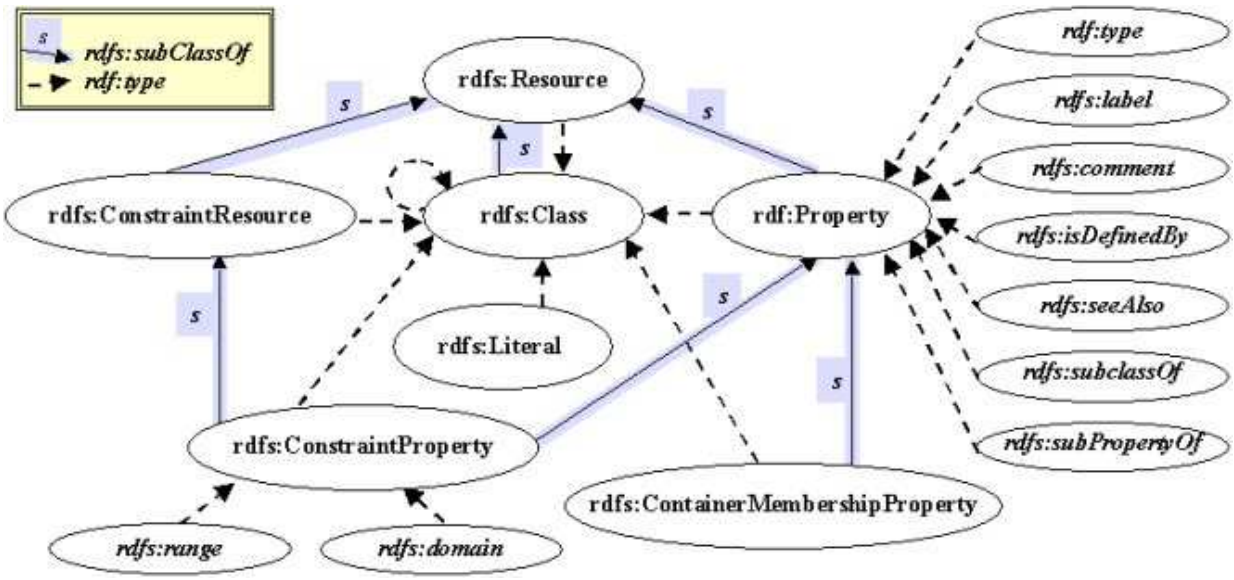
```
@prefix ABo: <ABoxLibros.n3#> .
 ABo:Boole      a ont:Persona,
                ...
 ABo:Yourcenar  a ont:Persona,
                ...
 <urn:isbn:0486600289>  a ont:DocPapel,
                          ont:Documento,
                          ont:Libro,
                        ont:autor ABo:Boole;
                        ...
 <urn:isbn:8489669643>  a ont:DocPapel,
                          ont:Documento,
                          ont:Ficcion,
                          ont:Historia,
                          ont:Libro,
                        ont:autor ABo:Yourcenar;
                        ...
```

RDFS: resumen gráfico (1)

De `http://www.w3.org/2001/sw/RDFCore/Schema/20010618/`



RDFS: resumen gráfico (2)



Píldora lógica: expresividad de RDF/RDFS

RDF/RDFS	DL	FOL
`:Rec rdf:type :Clase`	$Clase(REC)$	$c(R)$
`:Rec :Prop :Val`	$prop(REC,VAL)$	$p(R,V)$
`C1 rdfs:subClassOf C2`	$C_1 \sqsubseteq C_2$	$(\forall x)(c_1(x) \Rightarrow c_2(x))$
`P1 rdfs:subPropertyOf P2`	$P_1 \sqsubseteq P_2$	$(\forall x_1,x_2)(p_1(x_1,x_2) \Rightarrow p_2(x_1,x_2))$
`P rdfs:domain D`	$\exists P.\top \sqsubseteq D$	$(\forall x_1,x_2)(p(x_1,x_2) \Rightarrow d(x_1))$
`P rdfs:range R`	$\top \sqsubseteq \forall P.R$	$(\forall x_1,x_2)(p(x_1,x_2) \Rightarrow r(x_2))$

- RDFS permite expresar *algunos* axiomas (TBox)
- Carece de constructores de conceptos
- Como es un subconjunto de $\mathcal{FL}_0$ pueden aplicarse algoritmos de subsunción estructural
- Las únicas inferencias posibles son las derivadas de las jerarquías de clases y de roles y de las restricciones de dominio y rango.

Expresividad de RDF/RDFS en términos prácticos: ejemplo (1)

En una ontología sobre alumnos y nacionalidades *se puede expresar que:*

- hay clases, como `País`, `Persona`, `Alumno`, `Griego`, `Francés`...
- `Alumno` es una subclase de `Persona`
- `Eslovenia`, `Francia`, `España`... son ejemplares de `País`
- `nacionalidad` es una propiedad con dominio `Persona` y rango `País`
- `edad` es una propiedad con dominio `Persona` y rango `integer`
- `JJJ` es un ejemplar de la clase `Español` con `edad = 22`

Expresividad de RDF/RDFS en términos prácticos: ejemplo (2)

No se puede expresar que:

- `Persona` y `País` son clases disjuntas
- La clase `Europa` es la unión de las clases `España`, `Francia`...
- La clase `Alumno` está formada por los miembros de la clase `Persona` matriculados en al menos una asignatura.
- `ciudadano` es la propiedad inversa de `nacionalidad`
- la clase `Apátrida` está formada por los miembros de la clase `Persona` que no tienen valores de la propiedad `nacionalidad`
- la clase `Multinacional` está formada por los miembros de la clase `Persona` que tienen al menos dos valores de la propiedad `nacionalidad`
- la clase `Griego` está formada por los miembros de la clase la clase `Persona` que tiene `Grecia` como valor de la propiedad `nacionalidad`
- `edad` es un propiedad funcional

Expresividad de RDF/RDFS: «liberalidad»

Sin embargo...

Entre otras cosas (como la cosificación), las clases pueden ser miembros de otras clases:

`:Especie rdf:type rdfs:Class`

`:León rdf:type :Especie` $\rightsquigarrow E(L)$ en DL

(en lugar de `:León rdfs:SubClassOf :Especie` $\rightsquigarrow L \sqsubseteq E$)

`:leoVerdura rdf:type :León` $\rightsquigarrow L(LV)$

...

`:recurso1 rdf:type :recurso2`

`:recurso2 rdf:type :recurso3`

`...`

$\rightsquigarrow Rec3(Rec2(Rec1))$...