



ugr

Universidad
de Granada

DESARROLLO DE SISTEMAS DE SOFTWARE BASADOS EN
COMPONENTES Y SERVICIOS
MÁSTER EN INGENIERÍA INFORMÁTICA

PRÁCTICA 5

Consultas SPARQL

Autor:

Pablo Valenzuela Álvarez (pvalenzuela@correo.ugr.es)



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE TELECOMUNICACIÓN

Granada, 10 de enero de 2025

Índice

1. Cambios en la ontología	3
1.1. Clase Lector	3
1.2. Propiedades de objetos: compradoPor y haComprado	3
1.3. Propiedades de datos: tieneTitulo y tieneValoración	4
1.4. Ejemplos de uso	4
2. Consultas SPARQL	6
2.1. Consulta 1: Lectores que compraron un libro del género CienciaFiccion	6
2.2. Consulta 2: Títulos de libros del género NoFiccion	6
2.3. Consulta 3: Libros con valoración 4	7
2.4. Consulta 4: Libros comprados por cada lector	8
2.5. Consulta 5: Total de libros comprados	8
2.6. Consulta 6: Suma de los precios de libros comprados por cada lector	9
3. Repositorio GitHub	10

Índice de figuras

1.	Detalles de la clase Lector.	3
2.	Detalles de la propiedad de objeto compradoPor.	3
3.	Detalles de la propiedad de datos tieneTitulo.	4
4.	Detalles de la propiedad de datos tieneValoracion.	4
5.	Propiedades asociadas a el lector Yvanka.	4
6.	Propiedades asociadas al libro LaComunidadDelAnillo.	5
7.	Resultados de la consulta 1.	6
8.	Resultados de la consulta 2.	6
9.	Resultados de la consulta 3.	7
10.	Resultados de la consulta 4.	8
11.	Resultados de la consulta 5.	8
12.	Resultados de la consulta 6.	9

1. Cambios en la ontología

Para el correcto funcionamiento de esta práctica, hemos tenido que realizar pequeños cambios sobre la ontología realizada en la anterior práctica.

1.1. Clase Lector

El primer añadido, ha sido la creación de una clase para albergar lectores a la que hemos llamado *Lector*. Como se puede ver en la figura 1, hemos agregado algunas instancias y disjunciones con las demás clases primarias de la ontología.

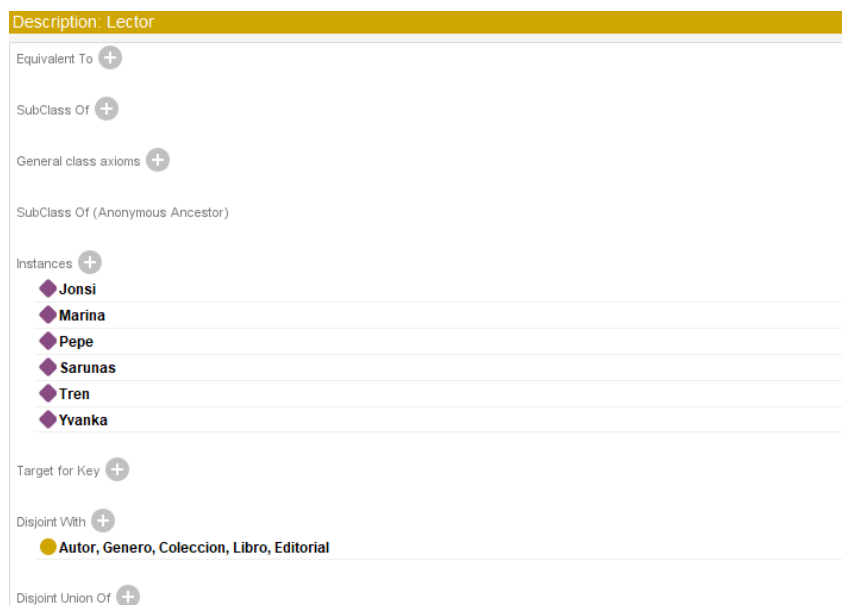


Figura 1: Detalles de la clase Lector.

1.2. Propiedades de objetos: compradoPor y haComprado

Acto seguido, se han añadido las propiedades de datos *compradoPor* y su inversa *haComprado*. Esta propiedad (como se ve en la figura 2) se aplica sobre el dominio *Libro* y su rango es *Lector*. Lógicamente, su inversa *haComprado*, invierte dominio y rango.

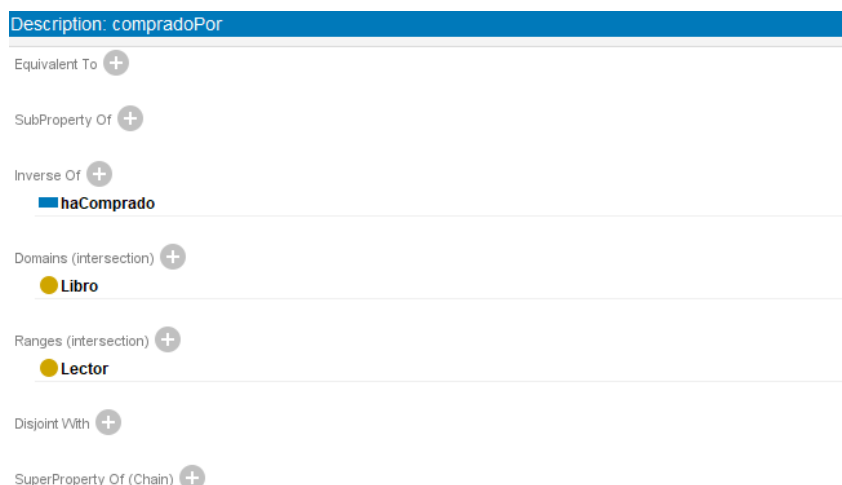


Figura 2: Detalles de la propiedad de objeto compradoPor.

1.3. Propiedades de datos: *tieneTitulo* y *tieneValoración*

Por último, hemos añadido dos propiedades de datos: *tieneTitulo* y *tieneValoracion*. Ambas se aplican sobre el dominio *Libro* y son usadas para dar un título al libro y dar una valoración numérica al libro (ver figuras 3 y 4).

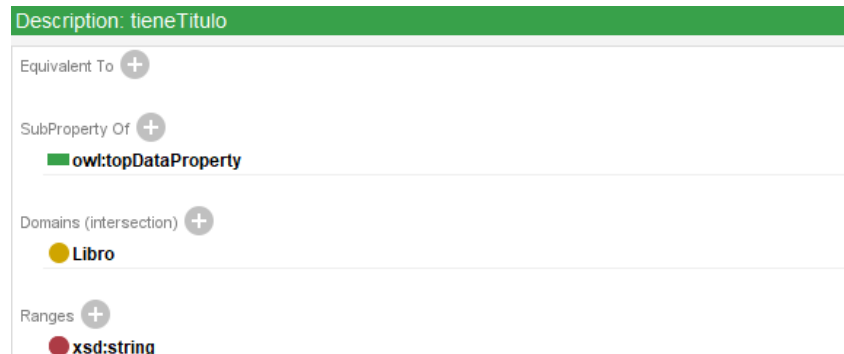


Figura 3: Detalles de la propiedad de datos *tieneTitulo*.

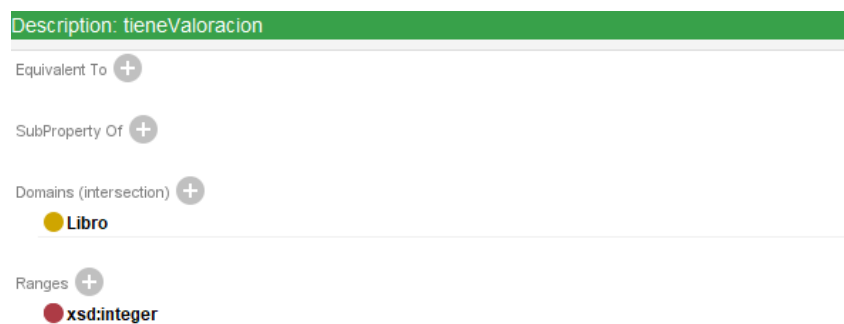


Figura 4: Detalles de la propiedad de datos *tieneValoracion*.

1.4. Ejemplos de uso

Las figuras 5 y 6 muestran un ejemplo de uso de las propiedades antes mencionadas, y también, de su aserción automática por parte de Protégé.

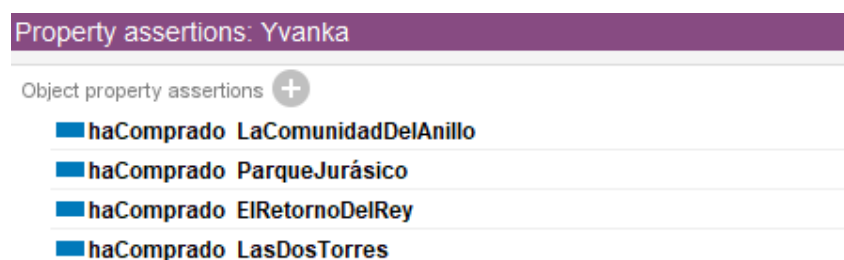


Figura 5: Propiedades asociadas a el lector *Yvanka*.


Property assertions: LaComunidadDelAnillo	
Object property assertions 	
publicadoPor	Minotauro
tieneGenero	Fantasia
escritoPor	JRRTolkien
perteneceColeccion	ElSeñorDeLosAnillos
compradoPor	Tren
compradoPor	Yvanka
Data property assertions 	
tieneTitulo	"La Comunidad del Anillo"
tieneAñoPublicacion	1954
tienePrecio	7.59
tieneValoracion	4

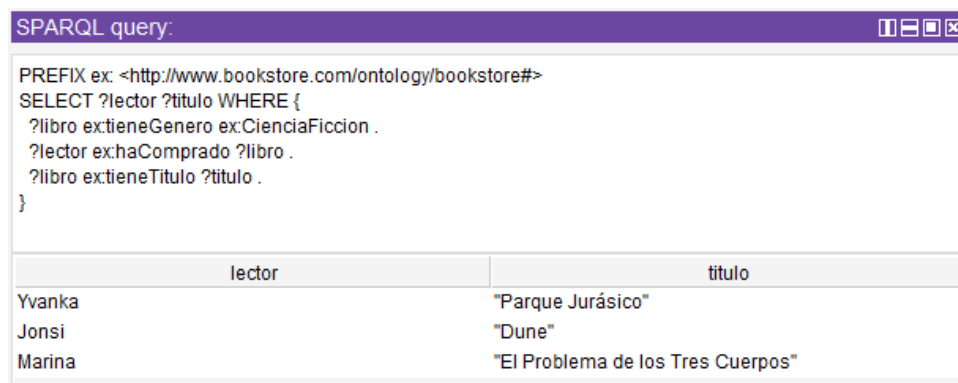
Figura 6: Propiedades asociadas al libro LaComunidadDelAnillo.

2. Consultas SPARQL

2.1. Consulta 1: Lectores que compraron un libro del género CienciaFiccion

```
PREFIX ex: <http://www.bookstore.com/ontology/bookstore#>
SELECT ?lector ?titulo WHERE {
  ?libro ex:tieneGenero ex:CienciaFiccion .
  ?lector ex:haComprado ?libro .
  ?libro ex:tieneTitulo ?titulo .
}
```

Como podemos observar en la figura 7, esta consulta muestra los clientes que han comprado algún libro del género “ciencia ficción”. Esto es útil si queremos ofrecer ofertas o descuentos a estos clientes ante la llegada de nuevos libros del mismo género.



The screenshot shows a SPARQL query interface with a purple header bar. The query is displayed in a text area, and the results are shown in a table below. The table has two columns: 'lector' and 'titulo'. The results are as follows:

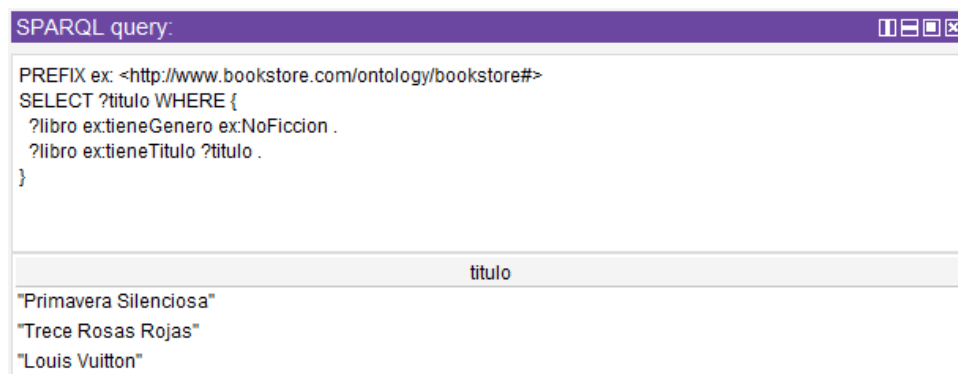
lector	titulo
Yvanka	"Parque Jurásico"
Jonsi	"Dune"
Marina	"El Problema de los Tres Cuerpos"

Figura 7: Resultados de la consulta 1.

2.2. Consulta 2: Títulos de libros del género NoFiccion

```
PREFIX ex: <http://www.bookstore.com/ontology/bookstore#>
SELECT ?titulo WHERE {
  ?libro ex:tieneGenero ex:NoFiccion .
  ?libro ex:tieneTitulo ?titulo .
}
```

La figura 8 muestra todos los títulos de los libros del género “no ficción”. Así podemos crear listas de los libros por género que haya disponibles en la librería.



The screenshot shows a SPARQL query interface with a purple header bar. The query is displayed in a text area, and the results are shown in a table below. The table has one column: 'titulo'. The results are as follows:

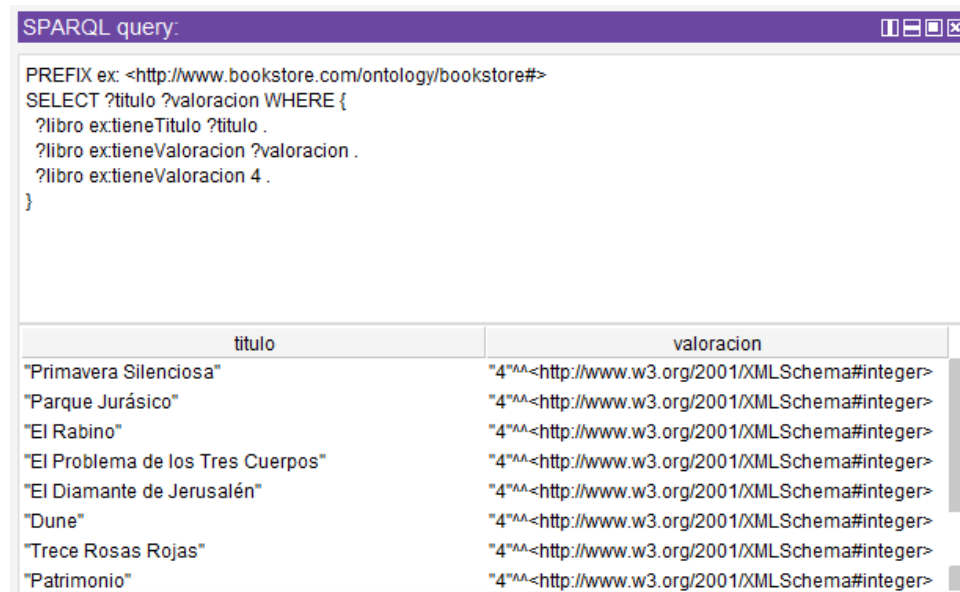
titulo
"Primavera Silenciosa"
"Trece Rosas Rojas"
"Louis Vuitton"

Figura 8: Resultados de la consulta 2.

2.3. Consulta 3: Libros con valoración 4

```
PREFIX ex: <http://www.bookstore.com/ontology/bookstore#>
SELECT ?titulo ?valoracion WHERE {
  ?libro ex:tieneTitulo ?titulo .
  ?libro ex:tieneValoracion ?valoracion .
  ?libro ex:tieneValoracion 4 .
}
```

Como se observa en la figura 9, se listan los libros con valoración 4. Podemos usar este tipo de consultas para crear listas de libros con mejor valoración.



The screenshot shows a web application window titled "SPARQL query:". The window has a purple header bar with standard window controls (minimize, maximize, close) on the right. The main area is divided into two sections. The top section contains the SPARQL query text, which is the same as the one in the previous block. The bottom section displays the results of the query in a table format. The table has two columns: "titulo" and "valoracion". The "titulo" column lists seven book titles, and the "valoracion" column shows the value "4" for each, followed by a URI in angle brackets: "<http://www.w3.org/2001/XMLSchema#integer>".

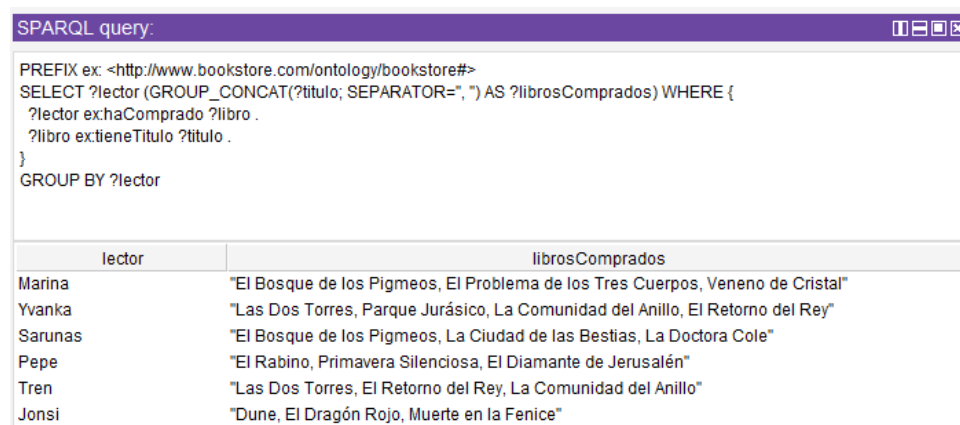
titulo	valoracion
"Primavera Silenciosa"	"4"<http://www.w3.org/2001/XMLSchema#integer>
"Parque Jurásico"	"4"<http://www.w3.org/2001/XMLSchema#integer>
"El Rabino"	"4"<http://www.w3.org/2001/XMLSchema#integer>
"El Problema de los Tres Cuerpos"	"4"<http://www.w3.org/2001/XMLSchema#integer>
"El Diamante de Jerusalén"	"4"<http://www.w3.org/2001/XMLSchema#integer>
"Dune"	"4"<http://www.w3.org/2001/XMLSchema#integer>
"Trece Rosas Rojas"	"4"<http://www.w3.org/2001/XMLSchema#integer>
"Patrimonio"	"4"<http://www.w3.org/2001/XMLSchema#integer>

Figura 9: Resultados de la consulta 3.

2.4. Consulta 4: Libros comprados por cada lector

```
PREFIX ex: <http://www.bookstore.com/ontology/bookstore#>
SELECT ?lector (GROUP_CONCAT(?titulo; SEPARATOR=", ") AS ?librosComprados) WHERE {
  ?lector ex:haComprado ?libro .
  ?libro ex:tieneTitulo ?titulo .
}
GROUP BY ?lector
```

Esta consulta muestra una lista agrupada por lector de todos los libros que han comprado (ver figura 10). Podemos usar esta información para hacer ofertas especiales a los clientes que mas compran.



The screenshot shows a SPARQL query interface with a purple header. The query is the same as in Figure 4. Below the query, a table displays the results. The table has two columns: 'lector' and 'librosComprados'. The data rows are as follows:

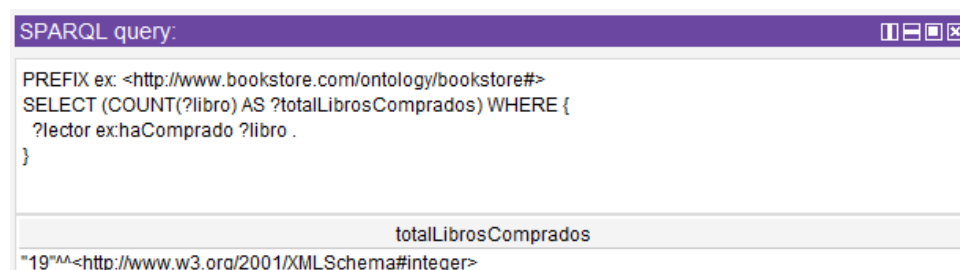
lector	librosComprados
Marina	"El Bosque de los Pigmeos, El Problema de los Tres Cuerpos, Veneno de Cristal"
Yvanka	"Las Dos Torres, Parque Jurásico, La Comunidad del Anillo, El Retorno del Rey"
Sarunas	"El Bosque de los Pigmeos, La Ciudad de las Bestias, La Doctora Cole"
Pepe	"El Rabino, Primavera Silenciosa, El Diamante de Jerusalén"
Tren	"Las Dos Torres, El Retorno del Rey, La Comunidad del Anillo"
Jonsi	"Dune, El Dragón Rojo, Muerte en la Fenice"

Figura 10: Resultados de la consulta 4.

2.5. Consulta 5: Total de libros comprados

```
PREFIX ex: <http://www.bookstore.com/ontology/bookstore#>
SELECT (COUNT(?libro) AS ?totalLibrosComprados) WHERE {
  ?lector ex:haComprado ?libro .
}
```

La figura 11 muestra el resultado de la consulta sobre mostrando la cantidad de libros que se han vendido.



The screenshot shows a SPARQL query interface with a purple header. The query is the same as in Figure 5. Below the query, a table displays the results. The table has one column: 'totalLibrosComprados'. The data row is as follows:

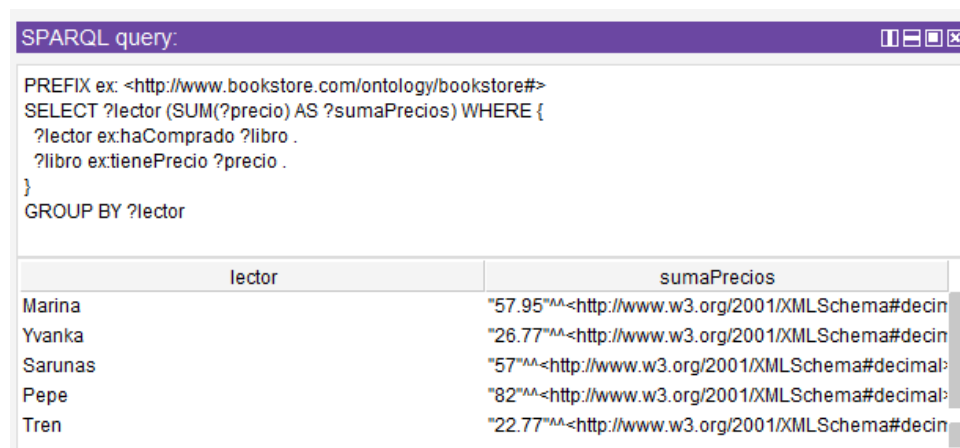
totalLibrosComprados
"19"^^<http://www.w3.org/2001/XMLSchema#integer>

Figura 11: Resultados de la consulta 5.

2.6. Consulta 6: Suma de los precios de libros comprados por cada lector

```
PREFIX ex: <http://www.bookstore.com/ontology/bookstore#>
SELECT ?lector (SUM(?precio) AS ?sumaPrecios) WHERE {
  ?lector ex:haComprado ?libro .
  ?libro ex:tienePrecio ?precio .
}
GROUP BY ?lector
```

Por último, la figura 12 muestra cuanto dinero ha gastado cada lector en la librería. Con la información de esta consulta podemos hacer algo similar a los expuesto en la consulta 4 (sección 2.4), y ofrecer descuentos a los que más han comprado.



The screenshot shows a SPARQL query interface with a purple header bar. The query is displayed in a text area, and the results are shown in a table below. The table has two columns: 'lector' and 'sumaPrecios'. The results list five readers: Marina, Yvanka, Sarunas, Pepe, and Tren, along with their total spending in a format that includes a decimal value and an XML Schema URI.

lector	sumaPrecios
Marina	"57.95"^^<http://www.w3.org/2001/XMLSchema#decimal>
Yvanka	"26.77"^^<http://www.w3.org/2001/XMLSchema#decimal>
Sarunas	"57"^^<http://www.w3.org/2001/XMLSchema#decimal>
Pepe	"82"^^<http://www.w3.org/2001/XMLSchema#decimal>
Tren	"22.77"^^<http://www.w3.org/2001/XMLSchema#decimal>

Figura 12: Resultados de la consulta 6.

3. Repositorio GitHub

Acceso al repositorio en GitHub de la asignatura: <https://github.com/Valenz23/DSS>