Facultad de
Comunicación y Documentación

UNIVERSIDAD DE GRANADA

GRADO EN INFORMACIÓN Y DOCUMENTACIÓN

TRABAJO FIN DE GRADO

LINKED DATA, UNA PROPUESTA PARA LA INTEROPERABILIDAD DE DATOS EN BIBLIOTECAS. EL CASO DE LA BIBLIOTECA DE LA UNIVERSIDAD DE GRANADA

Presentado por:

D. Wenceslao Arroyo Machado

Tutor:

Prof. Dr. José Antonio Senso Ruiz

Curso académico 2016 / 2017

D.: José Antonio Senso Ruiz, tutor del trabajo titulado **Linked Data, una propuesta para la interoperabilidad de datos en bibliotecas. El caso de la Biblioteca de la Universidad de Granada** realizado por el alumno **Wenceslao Arroyo Machado**, INFORMA que dicho trabajo cumple con los requisitos exigidos por el Reglamento sobre Trabajos Fin del Grado en *Información y Documentación* para su defensa.

Granada, _____ de _____ de _____

Fdo.: _____

Por la presente dejo constancia de ser el autor del trabajo titulado **Linked Data, una propuesta para la interoperabilidad de datos en bibliotecas. El caso de la Biblioteca de la Universidad de Granada** que presento para la materia Trabajo Fin de Grado del Grado en **Información y Documentación**, tutorizado por el profesor **José Antonio Senso Ruiz** durante el curso académico 2016-2017.

Asumo la originalidad del trabajo y declaro que no he utilizado fuentes (tablas, textos, imágenes, medios audiovisuales, datos y software) sin citar debidamente, quedando la Facultad de Comunicación y Documentación de la Universidad de Granada exenta de toda obligación al respecto.

Autorizo a la Facultad de Comunicación y Documentación a utilizar este material para ser consultado con fines docentes dado que constituyen ejercicios académicos de uso interno.

____ / ____ / ____

Fecha Firma

AGRADECIMIENTOS

“¡Libros! ¡Libros! He aquí una palabra mágica que equivale a decir: «amor, amor», y que debían los pueblos pedir como piden pan o como anhelan la lluvia para sus sementeras.” (García Lorca, 1996)

ÍNDICE

RESUMEN

Las bibliotecas se encuentran muy vinculadas al Linked Data debido al alto nivel de estructuración de sus datos, aunque los proyectos relacionados con ello son elaborados principalmente por grandes bibliotecas.

En el presente trabajo se ha determinado su estado de la cuestión, analizando algunos de los proyectos referentes, ciclos de vida y herramientas que intervienen durante el proceso, estableciendo tras ello una metodología y llevando acabo su implementación al completo, convirtiendo registros bibliotecarios en Linked Data, enriqueciéndolos por medio de otros conjuntos de datos y poniéndolos al alcance de todo el mundo.

De este modo, se ha realizado un estudio de caso usando para ello un conjunto de registros extraídos de la Biblioteca Universitaria de Granada con el fin de conocer, de primera mano, algunos de los problemas que se puede encontrar cualquier centro que desee convertir sus registros a Linked Data sin necesidad de tener que cambiar de sistema de automatización de bibliotecas.

ABSTRACT

Libraries are closely linked to Linked Data due to the high level of structuring of their data, although related projects are mainly developed by large libraries.

In the present work its status has been determined, analyzing some of the referring projects, life cycles and tools that intervene during the process, establishing afterwards a methodology and carrying out its complete implementation, converting library records into Linked Data, enriching them through other data sets and making them available to the world.

Thus, a case study has been carried out using a set of records extracted from the Library of the University of Granada in order to know, from first-hand, some of the problems that can find any center that wants to convert their records to Linked Data without having to change the library automation system.

1.- INTRODUCCIÓN

Desde que el primer sitio de la World Wide Web fuese publicado en agosto de 1991 hasta el día de hoy esta ha crecido a un ritmo vertiginoso, estimándose que actualmente existen 1,2 mil millones de sitios web (Internet Live Stats, 2017). Y es que según Cisco (2017), Internet se encuentra en la era del Zettabyte (ZB[1]) tras llegar en 2016 el tráfico global de Internet a 1.2 Zettabytes, esperándose que alcance los 3,3 Zettabytes en 2021.

A pesar de la inconmensurable cantidad de datos existentes al alcance de cualquier usuario de la Web, la mayoría se encuentran desordenados, desestructurados y con una escasa reutilización y relación entre ellos, más allá de los enlaces hipertextuales, que permiten desplazarse por la World Wide Web pero que carecen de semántica.

Tim Berners-Lee, creador de la Web, ya observó este problema, el cual afecta directamente a la recuperación de información, y enunció hace once años un método para solventarlo y favorecer la creación de la Web Semántica (Berners-Lee, 2006). Se trata del Linked Data o Datos Enlazados y su objetivo principal es facilitar datos bien definidos y estructurados, entendibles tanto por humanos como por máquinas, para ofrecer una nueva experiencia de navegación gracias a las relaciones entre ellos, fomentando la generación de nuevos contenidos y facilitando el acceso a la información. Todo ello bajo el supuesto de que la utilidad y calidad de los datos será mayor cuanto más interconectados se encuentren (Gómez Ruiz, 2013).

Este método de estructuración de datos no solo busca la publicación de datos en la Web sino la vinculación de los mismo, incrementando el crecimiento de la Web tanto a nivel de los documentos HTML (vista clásica de la Web) como a nivel de los datos expresados en RDF (vista de la Web Semántica) (W3C, 2017). De este modo, se pretende construir una red de datos donde los enlaces reflejan relaciones ente ellos.

En este marco, las bibliotecas, archivos y museos son también perjudicados por dicha falta de semántica y conexión, generándose mucha redundancia al tener que introducirse en extremo información ya publicada, al mismo tiempo que se echa en falta una mayor interoperabilidad entre ellos y con otros contenidos de la web que le permitan enriquecer su información.

Los documentos y metadatos que estas instituciones tienen entre sus manos, y a los cuales quieren facilitar su acceso y difusión, alcanzan un gran nivel de estructuración, suponiendo las bibliotecas, archivos y museos, en especial las primeras, un terreno idóneo para iniciativas

[1] 1 ZB = 10^{12} GB

como estas (Sulé, Centelles, Franganillo y Gascón, 2016). Son muchos los trabajos sobre estos proyectos (Vila-Suero, Villazon-Terrazas y Gomez-Perez, 2013; Deliot, 2014; Hallo, Lujan-Mora y Trujillo, 2014; Taylor, Jekjantuk, Mellish y Pan, 2014), así como los que analizan el estado de la cuestión del Linked Data en el mundo de las bibliotecas (Torre-Bastida, González-Rodríguez y Villar-Rodríguez, 2015; Smith-Yoshimura, 2016).

Tal es su importancia que ya en 2004 el W3C (World Wide Web Consortium) recomendó a las bibliotecas que publicasen sus datos utilizando tecnologías de la Web Semántica para incrementar su impacto digital y utilidad social (Hallo, Luján-Mora, Maté y Trujillo, 2016).

No obstante, no fue hasta 2006 cuando los principios fundamentales del Linked Data fueron enunciados por el padre de la World Wide Web, Tim Berners-Lee:

- Usar URIs para identificar las cosas
- Usar URIs HTTP
- Ofrecer información sobre los recursos usando RDF
- Incluir enlaces a otros URIs

Del mismo modo, y como actualización de lo anterior, Berners-Lee animó en 2010 a la comunidad, en especial a los proveedores de datos gubernamentales, a publicar y vincular Open Data (Datos Abiertos), tomando más fuerza el Linked Open Data (LOD) (Berners-Lee, 2006). Para ello estableció un sistema de estrellas, de modo que cuanto más abiertos y potentes sean los datasets[2] o conjuntos de datos elaborados que se publican y/o consumen con Linked Data más estrellas tendrán (Tabla 1), existiendo además otras propuestas que parten de esta como la de las siete estrellas (Hyvönen, Tuominen, Alonen y Mäkelä, 2014).

★☆☆☆☆	Disponible en la web, pero con una licencia Open Data
★★☆☆☆	Disponible de forma estructurada para ser leído por máquina
★★★☆☆	Disponible de forma estructurada para ser leído por máquina, y en formato no propietario
★★★★☆	Todo lo anterior, además de usar los estándares abiertos del W3C (RDF y SPARQL) para identificar cosas, de forma que los demás puedan enlazar a ellas
★★★★★	Todo lo anterior, además de enlazar tus datos con otros para dar contexto

Tabla 1. Rango de estrellas para el Linked Open Data

[2] Conjunto de datos estructurados que definen de manera inequívoca relaciones y propiedades entre diferentes clases bajo una misma temática.

Los datasets del Linked Open Data han experimentado un considerable crecimiento, tal y como muestra The Linking Open Data cloud diagram[3], ya que de los 203 conjuntos de datos que existían en 2010 se pasaron a 570 en 2014 y a 1.139 en 2017 (Ilustración 1), duplicándose de este modo durante los último tres años.

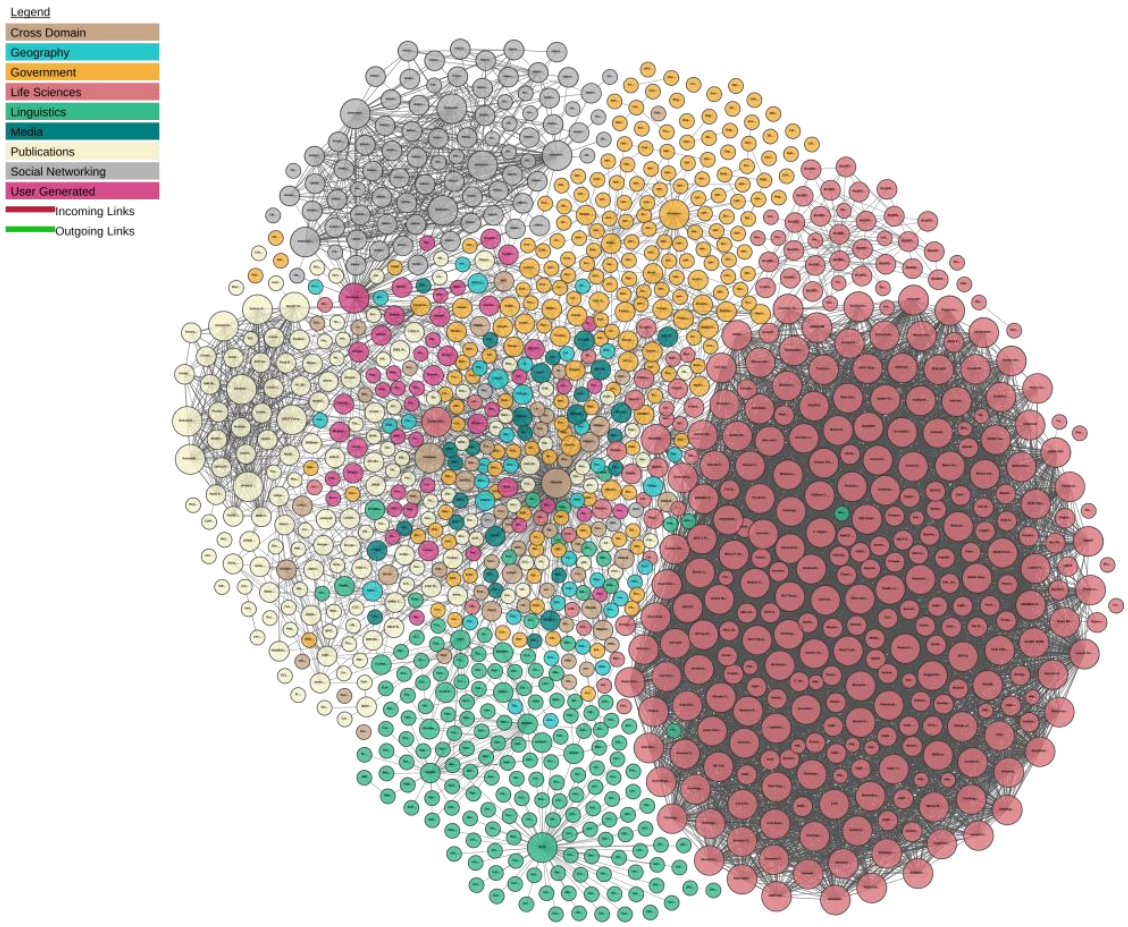

Ilustración 1. Diagrama de nube de puntos de los datasets Linked Open Data publicados hasta febrero de 2017. Disponible en: http://lod-cloud.net/ [Consultado en marzo de 2017]

Asimismo, su impacto y crecimiento también es claro en el mundo de las publicaciones científicas, donde no han dejado de aumentar los artículos sobre Linked Data y Linked Open Data, reflejando al mismo tiempo una clara relación con el mundo de las bibliotecas. En este aspecto, Estados Unidos, Alemania, Reino Unido y España se postulan además como los países con los que más colaboran en esta área a día de hoy (ANEXO A).

[3] http://lod-cloud.net/

1.1- LINKED DATA EN BIBLIOTECAS

En 2011 surge el W3C Library Linked Data Incubator Group “para ayudar a aumentar la interoperabilidad global de datos de la biblioteca en la Web”, que concluyó en 2011 (Bermès, Coyle y Dunsire, 2011), año en el que la Library of Congress anunció BIBFRAME (Bibliographic Framework), el cual se planteó como una evolución del formato MARC 21 a la Web Semántica y el Linked Data.

Es por todo ello que, gracias a la Web Semántica y concretamente el Linked Data, las bibliotecas se encuentran delante de una gran oportunidad para tomar parte activa en la misma y lograr una mayor visibilidad, ya que no les basta con estar solo en la Web sino que las bibliotecas tienen que ser la Web en sí misma (Hastings, 2015).

De entre las publicaciones que estudian el estado de la cuestión del Linked Data en bibliotecas, archivos y museos destaca la de Smith-Yoshimura (2016), por la amplitud e importancia de sus resultados y actualidad, de cuyos datos un 31 por ciento son pertenecientes a proyectos de bibliotecas universitarias, un 20 por ciento son de bibliotecas nacionales, un 6 por ciento de bibliotecas públicas, un 4 por ciento de museos y el resto otras instituciones como gobiernos.

En la línea de los resultados recogidos por dicho trabajo, la mayoría de dichos proyectos consumen y publican simultáneamente datos frente a los que simplemente consumen y los que solo publican, siendo estos últimos una clara minoría. Además, cerca de un 91 por ciento tienen por objetivo con todo ello exponer los datos a un público más amplio en la Web, al mismo tiempo que el 80 por ciento también pretende demostrar lo que puede hacerse con conjuntos de datos con Linked Data.

Ahora bien, el grueso de estas evaluaciones y estudios se localiza en el ciclo de vida de los diferentes proyectos a la hora de publicar y/o consumir Linked Data. En base a sus necesidades, cada proyecto utiliza o crea el ciclo de vida que mejor se adapte a sus objetivos, no obstante, la mayoría de estos comparten la necesidad de especificar, modelar y publicar los datos siguiendo los estándares de la Web (Hyland, Atemezing y Villazón-Terrazas, 2014).

Es por ello que existe un ciclo de vida común a la mayoría de los proyectos Linked Data y Linked Open Data que tengan por objetivo tanto la publicación como el enriquecimiento de los datos, y cuyas fases pueden ser variables en función de las necesidades concretas del mismo pero que en general son recurrentes (Ilustración 2).

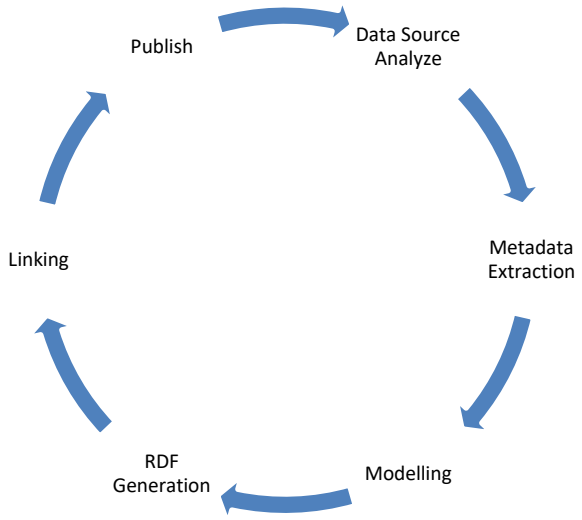

Ilustración 2. Ciclo de vida base de los proyectos Linked Open Data (Hallo, Lujan-Mora y Trujillo, 2014)

El objetivo y tendencias de las seis etapas del ciclo y sus diferentes subetapas, en base a los resultados recogidos por Smith-Yoshimura, son los siguientes:

1.1.1.- ANÁLISIS DE DATOS

El primer paso es la identificación de los datos que maneja la institución y que pueden ser reutilizados por otros, siendo los principales datos publicados por las instituciones estudiadas bibliográficos (56%), autoridades (45%), metadatos (43%) y ontologías/vocabularios (30%). Por su parte, sus subetapas se dividen en:

1. Identificar la fuente y los atributos de los datos a publicar y que serán enlazados con otros datasets o conjuntos de datos.
2. Analizar la fuente de los datos o, lo que es lo mismo, el software en el que están almacenados, el formato y la base de datos en la que se encuentra.
3. Identificar las procedencias y licencias de los datos para evitar problemas de carácter legal con la posterior publicación de los mismos.

1.1.2.- EXTRACCIÓN DE DATOS

Tras ello se extraen los datos de la fuente original, pudiendo almacenarlos en una base de datos intermedia de cara a su limpieza. Para ello los principales pasos son:

1. Extracción y almacenamiento de los metadatos.
2. Limpieza de los metadatos para solucionar la ausencia de valores o errores de los mismos.

1.1.3.- MODELADO

Con los datos ya en nuestras manos, se diseña un vocabulario para poder describir correctamente los datos extraídos en formato RDF. Para ello hay que considerar las siguientes subetapas:

1. Selección de vocabularios/ontologías. Es posible encontrar una gran cantidad de vocabularios y ontologías, usándose varios de manera conjunta. Entre los que más suelen encontrarse están Dublin Core, BIBO, BIO, FOAF, FRBR, FRAD, FRSAD, IFLA, ISBD, INTERMARC, MADS/RDF, XML-EAD, OAI_ORE, RDF, RDF Schema ORG, OWL, RDA, SKOS, WGS84 o EVENT. No obstante, los vocabularios y ontologías que más fueron utilizados en 2015, por orden de mayor a menor frecuencia, fueron:
    a. Simple Knowledge Organization System (SKOS)
    b. Friend of a Friend (FOAF)
    c. DCMI Metadata Terms (dcterms)
    d. Dublin Core Metadata Element Set (dce)
    e. Schema.org vocabulary (schema)
    f. The Bibliographic Ontology (bibo)
    g. Vocabulario local
    h. VOCABS rda
    i. Europeana Data Model vocabulary (edm)
    j. ISBD elements (isbd)
    k. WGS84 Geo Positioning (geo)
    l. BIBFRAME Vocabulary (bf)
2. Desarrollo de este.
3. Validación para evitar la existencia de fallos.
4. Selección de la licencia.

Independientemente de los vocabularios y ontologías que se usen, un aspecto clave en su publicación y consumo son las licencias. Para que otras personas puedan utilizar los datos publicados con una base legal segura es imprescindible que estos indiquen explícitamente qué licencia se aplica a ellos. En caso de no especificarse ninguna, las personas no pueden utilizar sus datos en aplicaciones importantes (W3C, 2011).

Como anteriormente se mencionó, dentro del sistema de estrellas establecido a la hora de publicar Linked Open Data, el primer requisito es que se establezca una licencia Open Data

(Datos Abiertos), el cual es un movimiento digital, al que se están apuntando cada vez más gobiernos e instituciones, y que promueve la publicación de datos libres de restricciones de copyright, patentes u otros, y en formatos que permitan su reutilización con independencia del fin que se le dé (Biblioteca del Congreso Nacional de Chile, 2012).

Por ello es fundamental diferenciar entre Linked Data y Linked Open Data y establecer una licencia adecuada, así como considerar esta antes de usar los datos publicados por otros. En el año 2015 las licencias más usadas por los proyectos fueron las siguientes:

1. CC0 1.0 Universal – Es la licencia más empleada y supone una renuncia de todos los derechos por parte del autor sobre los datos publicados.
2. Open Data Commons Attribution (ODC-BY) – Las personas que hagan uso de datos bajo esta licencia deben reconocer su procedencia, pero pueden usarlos y distribuirlos sin ninguna limitación más.
3. Open Data Commons Open Database License (ODC-ODbl) – No solo es necesario reconocer su procedencia, sino que la difusión que se le puedan dar a productos derivados de estos datos deben tener la misma licencia.
4. Public Domain Dedication and License or PPDL – Se renuncia a los derechos sobre los datos y estos pasan a dominio público, del mismo modo que con CC0 1.0 Universal.
5. Creative Commons Attribution-NonCommercial-NoDerivatives (BY-NC-ND) – Es necesario reconocer la procedencia y no se puede compartir los datos bajo ninguna modificación o uso comercial.

1.1.4.- GENERACIÓN DE RDF

Realizada ya la modelación, le llega el turno a la generación del RDF, el cual pasa por las siguientes subetapas:

1. Selección de tecnologías para la generación de RDF.
2. Transformación de los datos originales a RDF.
3. Generación de RDF.

Las sintaxis más utilizadas, también en 2015, para la serialización – codificación de las tripletas que componen un grafo RDF en un formato determinado – fueron:

1. RDF/XML – Es el formato básico y más utilizado con diferencia. Serializa grafos RDF por medio de sintaxis XML. Durante años fue la única para RDF (Manola, Miller y McBride, 2014).

2. Terse RDF Triple Language (Turtle) – Su uso es más cómodo y equilibrado ya que permite que los grafos RDF estén completamente escritos en un formato de texto compacto y natural, con abreviaturas para patrones de uso comunes y tipos de datos (Beckett, Berners-Lee, Prud'hommeaux y Carothers, 2014).
3. JSON-LD – Sintaxis ligera para serializar Linked Data en JSON. Está destinado a ser una forma de utilizar Linked Data en entornos de programación basados en web, crear servicios web interoperables y almacenar datos vinculados en motores de almacenamiento basados en JSON (Sporny et al., 2014).
4. N-Triples – Serialización de RDF en texto plano derivado de Turtle. Su intención original era para usos de prueba, pero su éxito lo ha transformado en un formato popular, aunque farragoso para grandes cantidades de datos (Beckett, 2014).
5. RDFa Core – Su uso es idóneo en los casos en que la obtención de los datos se realiza incrustados en ficheros HTML (Adida, Birbeck, McCarron y Herman, 2015).

1.1.5.- ENLAZADO

Una vez se tienen correctamente definidos los datos es el momento de enlazar con otros datasets de cara a enriquecerlos. Para ello hay que considerar las siguientes subetapas:

1. Buscar conjuntos de datos con los que enlazar, siendo Datahub[4] una de las principales herramientas para ello.
2. Enlazar con datasets externos relacionados.
3. Verificar las relaciones.

Cabe destacar que los datasets más consumidos fueron los siguientes, de los cuales salvo DBpedia y GeoNames son todos proyectos relacionados con bibliotecas:

1. Virtual International Authority File (VIAF)
2. DBpedia
3. GeoNames
4. id.loc.gov
5. Los propios recursos de los encuestados
6. Getty's Art and Architecture Thesaurus
7. FAST Linked Data
8. WorldCat.org
9. data.bnf.fr

[4] https://datahub.io/

1.1.6.- PUBLICACIÓN

El siguiente y último paso es publicar los datasets elaborados en RDF siguiendo los principios del Linked Data. Las subetapas necesarias son las siguientes:

- Publicar el dataset y vocabulario.
- Definir y publicar los metadatos.

Para ello hay una gran cantidad de tecnologías disponibles, siendo las usadas por proyectos relacionados con bibliotecas, archivos y museos muy variadas (Tabla 2), sin un único y claro ganador:

Número de proyectos	Tecnologías
10 o más	SPARQL, Java, XSLT, Zorba
2-9	Solr, Virtuoso Universal Server, Google Refine, Jena Applications, RDF Store, Drupal7, Python, Apache Fuseki, ElasticSearch, Perl, Metafacture, DIGIBIB, MongoDB, 4store, Apache Marmotta, BlazeGraph, GraphDB, Hydra, Numishare, Rails
1	Apache Tomcat, Cubicweb, Django, dotNetRDF, Fedora Commons, Hbase/Hadoop, JAX-RS, Joomla, LibHub, Mapping Memory Mapper (3M), MARC Report and MARC Global, MySQL, Node.js, Oracle, PoolParty, Protègè, Pubby, r2rml-parses, Ruby, Ruby Virtuoso triplestore, Sesame, skosmos, Wordpress

Tabla 2. Tecnologías empleadas para la publicación de Linked Data (Smith-Yoshimura, 2016)

En cuanto a los métodos preferidos de cara a su acceso, están los siguientes, siendo el primero de ellos el favorito de manera muy clara respecto al resto:

1. Páginas web
2. Negociación de contenido
3. File dumps o archivos de volcado
4. SPARQL Endpoint
5. Editor de SPARQL
6. Aplicaciones

Por último, las principales barreras descritas por las instituciones al cargo de dichos proyectos para su publicación son:

1. La curva de aprendizaje para el personal.
2. La inconsistencia de los datos heredados.
3. Selección de ontologías apropiadas para representar los datos.
4. El establecimiento de los enlaces.
5. Poca documentación o asesoramiento sobre cómo construir los sistemas.
6. La falta de herramientas.
7. Software inmaduros.
8. Determinar quién posee los datos.

1.2- BUENAS PRÁCTICAS

Por otra parte, el W3C (Lóscio, Burle y Calegari, 2016) ha identificado los principales desafíos a los cuales se enfrentan todos aquellos que consumen o publican sus datos en la Web, para los cuales se han asignado una o varias propuestas de buenas prácticas (Best Practices), destinadas a dar respuesta a cada uno de ellos (Ilustración 19).

Los beneficios que estas aportan, constituyendo los dos primeros (Reutilización y Confianza) las principales inclinaciones de las buenas prácticas (Sánchez, 2016), son los siguientes:

- Reutilización – Mejorar las posibilidades de que el dataset sea reutilizado.
- Confianza – Aumentar la confianza del dataset de cara a consumidores y usuarios.
- Comprensión – Incrementar la compresión de los datos y metadatos.
- Acceso – Proporcionar diferentes métodos de acceso a los datos.
- Enlazabilidad – Establecer enlaces entre recursos de datos (datos y datasets).
- Interporabilidad – Mejorar la relación entre desarrolladores, editores y consumidores.
- Detectabildiad – Facilitar el descubrimiento automático de datos y datasets.
- Procesabilidad – Procesamiento y manipulación automática de conjuntos de datos.

2.- OBJETIVOS

El objetivo general consiste en establecer una metodología para convertir los registros bibliotecarios en Linked Data, enriquecerlos por medio de otros conjuntos de datos y ponerlos al alcance de todo el mundo.

Para ello, se han establecido una serie de objetivos específicos que pasan por:

- Elaborar el estado de la cuestión del Linked Data y su aplicación en bibliotecas.
- Diseñar un modelo.
- Seleccionar las herramientas Linked Data que puedan servir para el modelo diseñado.
- Estudiar cómo enriquecer los registros bibliográficos.
- Implementación de la metodología diseñada.
- Determinar las dificultades para el tratamiento de los registros bibliográficos en la aplicación del Linked Data.

3.- METODOLOGÍA

El trabajo se ha estructurado en torno a cuatro grandes fases consecutivas:

1. Primera fase – Revisión bibliográfica para determinar el estado de la cuestión del Linked Data en bibliotecas, prestando atención a los principales proyectos.
2. Segunda fase – Estudio de los requisitos y principales herramientas en cada una de las etapas del ciclo de vida para convertir los registros bibliográficos en Linked Data.
3. Tercera fase – Diseño de un modelo para llevar a cabo todo el proceso.
4. Cuarta fase – Implementación del modelo establecido al completo, determinando los principales problemas encontrados durante su elaboración y soluciones a los mismos, existentes y que se han tomado para llevarlo a cabo.

Con el fin de conocer las dificultades más comunes a las que se van a enfrentar la mayoría de las bibliotecas, se han empleado los registros disponibles en la Biblioteca de la Universidad de Granada, usando para ello todos los formatos ofrecidos, los cuales se estudiarán a fin de conocer el idóneo para extraer de la mejor forma posible los datos referidos al autor, título, publicación, materias e ISBN, con los cuales se trabajarán durante el proyecto. Debido al tamaño del catálogo, se seleccionará una muestra de los mismos no inferior a los 1.000 registros y bajo una misma temática. De este modo, y aparte de ello, no se parte con ninguna predisposición a utilizar una metodología ni herramientas concretas, sino que será tras las iniciales evaluaciones cuando se determinarán, contemplando así todos los caminos posibles.

4.- DESARROLLO

4.1. PRINCIPALES PROYECTOS DE BIBLIOTECAS

Al ocupar las bibliotecas nacionales y algunos proyectos como Europeana un papel destacado en el ámbito del Linked Data y las bibliotecas, se ha realizado una evaluación de algunos de ellos a fin de determinar cuáles son los pasos que dan en común, así como sus diferencias, para establecer una metodología válida.

Los proyectos elegidos han sido los llevados a cabo por la Biblioteca Nacional de España (BNE), British Library (BL), Bibliothèque nationale de France (BnF), Europeana y Library of Congress (LoC). Su elección se ha realizado en base a su coincidencia en diferentes publicaciones enfocadas al análisis de la situación actual del Linked Data en bibliotecas (Papadakis, Kyprianos y Stefanidakis, 2015; Torre-Bastida, González-Rodríguez y Villar-Rodríguez, 2015; Hallo et al., 2016), así como en estudios de caso (Deliot, 2014; Vila-Suero, Villazon-Terrazas y Gomez-Perez, 2013; Hallo, Lujan-Mora y Trujillo, 2014; Wenz, 2013). Es por ello que se prestará atención a dichos proyectos, los cuales se presentan como referentes y están todos enfocados al Linked Open Data, para desarrollar una metodología que permita transformar los registros bibliográficos en MARC 21 de cualquier biblioteca en Linked Data.

En primer lugar, y antes de la evaluación, hay que destacar los beneficios que el Linked Data aporta a las bibliotecas. Y es que los responsables al cargo de dichos proyectos afirman que estos pasan por:

- Mejorar la visibilidad de los datos.
- Reaprovechar los datos de libros publicados por otros y añadirles los propios (como su disponibilidad).
- Establecer vínculos con otros servicios y favorecer el desarrollo de mashups.
- Mejora la recuperación de datos abiertos.
- La interoperabilidad conseguida no afecta a los modelos de la fuente de datos.
- Es posible consultar los metadatos vinculados a partir de múltiples instituciones.
- Se permite el modelado de las “cosas de interés” relacionadas con un recurso bibliográfico, como personas, lugares, eventos y temas.

Ahora bien, analizando cada uno de esos proyectos es posible observar algunas diferencias a la hora transformar y publicar sus registros en Linked Data.

4.1.1.- BNE (DATOS.BNE.ES)

La Biblioteca Nacional de España ofrece acceso al catálogo bibliográfico y de autoridades como Linked Open Data.

Han transformado los registros de MARC 21 a RDF por medio de un proceso que se ha automatizado en parte con el software Marimba, permitiendo además el descubrimiento de enlaces hacia otros datasets, y ayudándose en esta parte también por medio de otro software, Silk.

La publicación y consulta de sus datos es posible gracias al repositorio RDF Virtuoso y la interfaz para SPARQL Endpoint, pubby (Torre-Bastida, González-Rodríguez y Villar-Rodríguez, 2015).

SPARQL (SPARQL Protocol and RDF Query Language) es un lenguaje de consulta de grafos del mismo modo que SQL (Structured Query Language) lo es con las bases de datos relacionales.

El ciclo de vida de la Biblioteca Nacional de España utilizado para la publicación de sus datos mediante Linked Data (Vila-Suero, Villazon-Terrazas y Gomez-Perez, 2013) está compuesto por siete pasos (Tabla 8). Las diferencias más notables encontradas en este apartado es que uno de ellos está directamente centrado en la limpieza de los datos y el último en el desarrollo de aplicaciones.

4.1.2.- BRITISH LIBRARY (BNB.DATA.BL.UK)

La British Library ofrece acceso al British National Bibliography (BNB) como Linked Open Data. No lleva a cabo una transformación directamente de sus registros en MARC 21 a RDF, sino que en un primer lugar identifica sus “objetos de interés”, lo que el registro del catálogo dice sobre "las cosas en el mundo”, incluyendo esto conceptos y abstracciones, así como objetos materiales. Estas entidades o clases son identificadas mediante URIs, creadas por ellos. Tras esto, se describen las clases y sus relaciones entre sí, para lo cual definieron sus propias clases y propiedades, documentadas en el British Library Terms RDF schema (Deliot, 2014).

Los datos enlazados del BNB siguen dos modelos claramente diferenciados dependiendo de si están referidos a libros o publicaciones seriadas, encontrándose por lo tanto el “British Library data model for books” y “British Library data model for serials”.

4.1.3.- BIBLIOTHÈQUE NATIONALE DE FRANCE (DATA.BNF.FR)

Su flujo de trabajo comienza recogiendo sus datos tal cual se encuentran, para mantener el proceso de producción de datos globales y sus catálogos existentes. Mantienen separada la base de archivo de la base bibliográfica, y muestran datos en la Web con vocabularios comúnmente utilizados.

La Bibliothèque nationale de France ha trabajado con diferentes bases de datos, vinculando metadatos de documentos en papel con su versión digitalizada o recopilando archivos con documentos publicados. Han transformado los datos de bases de datos no interoperables en datos estructurados e intercambiables compatibles con los estándares de la Web Semántica.

A los recursos producidos por la BnF se les asigna un identificador permanente, un identificador Archival Resource Key (ARK). Estos recursos son registros de autoridad y catálogo, que sirven de ayuda para la localización de archivos, manuscritos y documentos digitales (Wenz, 2013).

Los identificadores ARK son un sistema de localización y protocolo de forma independiente, que permiten la identificación permanente, salvaguardando los recursos y sus sistemas de denominación a través del tiempo. Se trata pues de una URL efectiva, especialmente configurada y única a nivel mundial (IFLA, 2014).

El sistema está basado en tres requisitos:

- Un enlace desde el objeto a un compromiso para la custodia.
- Un enlace desde el objeto a los metadatos que lo describen.
- Un enlace al objeto en sí mismo (o sustituto adecuado).

4.1.4.- EUROPEANA (DATA.EUROPEANA.EU)

Europeana sigue el modelo EDM (Europeana Data Model), que estructura y representa los datos con los que las instituciones de patrimonio cultural contribuyen a Europeana. Este modelo ofrece mayor expresividad y flexibilidad que el modelo ESE (Europeana Semantic Elements), al cual sustituye.

EDM incluye conexiones semánticas a fuentes externas y reutiliza elementos procedentes de vocabularios ya establecidos, como Dublin Core, OAI-ORE, SKOS y CIDOC-CRM, reduciendo el coste de su creación.

Los requisitos principales de EDM incluyen:

- Distinguir entre un "artículo suministrado" (por ejemplo un libro) y sus representaciones digitales.
- Distinguir entre un elemento y el registro de metadatos que lo describe.
- Permitir la ingestión de múltiples registros para el mismo artículo, que puede contener declaraciones contradictorias sobre ella.

Es por ello que los datos que se pueden encontrar para cada clase de recurso son la propia representación del objeto, sus datos descriptivos, los datos referentes al proveedor y metadatos descriptivos asignados por este, y los metadatos asignados por Europeana (2015).

4.1.5.- LIBRARY OF CONGRESS (ID.LOC.GOV)

En su caso, la Library of Congress proporciona a través del Linked Data acceso a sus estándares y vocabularios, se encuentra vinculada a la BnF y ha impulsado el modelo BIBFRAME (Bibliographic Framework).

Ahora bien, observando de manera general los vocabularios y ontologías usados por estos proyectos (Tabla 9), es posible apreciar que Dublin Core es el único usado por todos ellos, así como que SKOS es el siguiente más empleado, existiendo en todas las demás menos coincidencias.

Otro aspecto interesante es la forma a través de la cual publican sus datos los diferentes proyectos (Tabla 3), en los que destaca Europeana por ser la única que lo hace de las tres formas analizadas.

Proyectos	SPARQL Endpoint	Dump files	URIs desreferenciables[5]
British Library	✓		
BnF		✓	✓
BNE	✓		
LoC		✓	✓
Europeana	✓	✓	✓

Tabla 3. Formas de publicación de Linked Data de los proyectos más destacados del ámbito bibliotecario (Papadakis, Kyprianos y Stefanidakis, 2015)

4.2. PRINCPALES HERRAMIENTAS

Para la selección de las herramientas necesarias para trabajar con Linked Data se ha llevado a cabo una búsqueda y análisis descriptivo de las principales disponibles para ello (AKSW,

[5] Una URI desreferenciable posibilita el acceso a la descripción del recurso identificado por ella a través del protocolo HTTP (Papadakis, Kyprianos y Stefanidakis, 2015)

2016; Import.io, 2015; Bustán, María y González, 2016; Hallo, Lujan-Mora y Trujillo, 2014), organizándolas en base a su función principal y fase del ciclo de vida del Linked Data aplicable para su posterior uso. Pese a su importancia para el desarrollo del proyecto, por problemas de espacio se encuentra en el anexo C.

4.3. PROPUESTA DE METODOLOGÍA

Tras una exhaustiva revisión de los principales proyectos Linked Data en bibliotecas y su estado actual en dicho ámbito, destacando en este sentido además la hoja de ruta establecida para dicha transición por el proyecto BIBLOW[6] (MacKenzie., Carl, Stahmer y Gloria, 2017), se ha formulado una propuesta metodológica (Tabla 4) resultado de combinar la seguida por la BNE *(Vila-Suero, Villazon-Terrazas y Gomez-Perez, 2013)* y la de proyectos Linked Open Data *(Hallo, Lujan-Mora y Trujillo, 2014)*. Se define así un proceso en el que, a partir de unos registros bibliográficos tomados de la Biblioteca de la Universidad de Granada, se busca su conversión y enriquecimiento a través del Linked Data para finalmente publicarlos.

Etapa	*Descripción*	*Tareas*
1. Determinar	Identificación y descripción de los datos.	a. Identificar y analizar los datos y fuente de datos (software, formato, base de datos…) b. Identificar su licencia c. Determinar una licencia
2. Limpiar	Almacenamiento y corrección de los datos.	a. Data curation
3. Modelar	Desarrollo de un vocabulario para describir los datos en formato RDF.	a. Seleccionar los vocabularios b. Creación de mapa c. Asignar URIs
4. Generar	Generación de los recursos RDF.	d. Seleccionar las tecnologías para la generación de RDF e. Transformar los datos fuente en RDF f. Validarlo
5. Enlazar	Conectar el dataset a otros que lo enriquezca.	a. Buscar datasets relevantes b. Descubrir relaciones c. Enlazar d. Verificar los enlaces
6. Publicar	Publicación del dataset	a. Escoger el formato y plataforma b. Publicar el dataset c. Publicar sus metadatos

Tabla 4. Propuesta de metodología para publicar registros bibliográficos como Linked Data

[6] https://bibflow.library.ucdavis.edu/

4.4. IMPLEMENTACIÓN

4.4.1.- ETAPA 1: DETERMINAR LOS DATOS

En esta primera parte se han identificado y reunido los datos a emplear. Estos son los disponibles en los registros almacenados en el catálogo general de la Biblioteca de la Universidad de Granada. Los datos que se buscan extraer, de la mejor forma posible para minimizar después la etapa de limpieza de datos (Data Curation) y facilitar su carga, son los referentes al autor, título, publicación, materias e ISBN. Es por ello que la planificación de esta tarea se divide en cuatro pasos (Ilustración 3), explorándose en cada uno de ellos todos los caminos posibles para garantizar que estos se recuperan en el mejor estado posible.

Ilustración 3. Pasos para la tarea de identificación de los datos

Para su descarga, la Biblioteca de la Universidad de Granada dispone de una base de datos en la que se encuentran almacenados todos los registros bibliográficos. El acceso y búsqueda de los mismos puede hacerse a través de dos formas distintas:

- Catálogo web – Por medio dos motores de búsqueda diferentes:
  - Adrastea[7]
  - VELETA[8]
- Servidor Z39.50 de la Biblioteca de la Universidad de Granada[9]

De este modo, en primer lugar se optó por el catálogo web. Debido a la elevada cantidad de registros disponibles, localizados en VELETA (un total de 1.359.685 registros a fecha de enero de 2017), se ha seleccionado una muestra para trabajar sobre ellos de manera ágil y poder llevar el resto de procesos de una manera suficientemente representativa. Por consiguiente, los registros que se van a emplear son los recogidos como resultado de la búsqueda de todos los que contengan como materia “documentación” o “biblioteconomía”.

Curiosamente, se recuperaron 1.251 registros en Adrastea y 1.249 en VELETA, diferencia que se debía a dos registros que incluían las materias sin acentuar y que dicho buscador no encontraba ya que, al contrario que el otro, no parece aplicar lematizadores.

Una vez ya localizados todos, estos fueron guardados a través del catálogo tradicional,

[7] http://adrastea.ugr.es
[8] http://bencore.ugr.es
[9] http://www.ugr.es/~biblio/ayudaopac/z39.htm

Adrastea, ya que es el único que permite hacerlo de manera masiva y exportarlos, siendo estos todos los formatos posibles que ofrece y en los que fue llevado a cabo:

- Pantalla Completa
- Presentación Abreviada
- ProCite
- Endnote-Refworks
- MARC

En relación con este último, y de manera paralela a las búsquedas del catálogo, se intentaron descargar todos esos mismos registros vía servidor Z39.50, ya que a través de él se pueden obtener en formato MARC. Para ello se empleó la misma consulta que antes y se lanzó al servidor a través de la aplicación web de la Library of Congress[10], las aplicaciones de escritorio MarcEdit 6[11] y BibDataZU Z39.50 Client[12], e incluso Refworks.

Sin embargo, hubo varios problemas. El primero es que la Biblioteca de la Universidad de Granada tiene limitado su acceso exclusivamente a 500 registros, Refworks puede superarlo pero no ofrece los registros en MARC sino como registros bibliográficos en RIS y Bibtex. Se intentó verificar si se podía superar dicho límite, para lo cual se contactó con un bibliotecario a través del servicio "Sesión Online con la Biblioteca" [13], el cual confirmó que no había forma de sobrepasarlo. También se pensó en lanzar dos consultas, una para cada una de las dos palabras clave, ya que estas están unidas por condicional booleano de unión (OR), pero ambos ofrecen más de 500 registros como resultado. Es por ello que dicha vía quedó descartada por limitaciones de la Biblioteca de la Universidad de Granada.

Por su parte, Adrastea no hace tampoco una correcta exportación en dicho formato. Tanto al descargar los 1.251 como 50 de ellos, siempre aparecían algunos con campos vacíos o errores que impedían su visualización. Este error se solucionó a través de Notepad++ 7.3.1[14]. De un vistazo se descubrió que el fallo de dicho archivo se encontraba en varios saltos de línea que aparecen al llega al carácter 40.000, impidiendo con ello una correcta lectura. Por ello, se eliminaron dichos saltos, quedando todo el archivo reducido a una única línea, y se guardó. Aclarar también que desde el servidor Z39.50 se obtiene un archivo ISO 2709, extensión .mrc (Norma UNE-ISO 2709:2006), mientras que desde Adrastea es en MARC, extensión .txt.

[10] https://www.loc.gov/z3950/gateway.html
[11] http://marcedit.reeset.net/
[12] http://bibdata.com/
[13] http://biblioteca.ugr.es/pages/biblioteca_responde/sesiononline
[14] https://notepad-plus-plus.org/download/v7.3.1.html

Con todos los ficheros descargados en perfecto estado, se ha llevado a cabo un análisis del contenido y estructura de todos ellos, buscándose en este sentido además el formato más manejable para su posterior limpieza, concretamente CSV (Comma-Separated Values), TSV (Tab-Separated Values) u otro con una delimitación similar que permita separar los registros por filas y los campos por columnas:

- MARC – A través de MarcEdit 6 se puede exportar su contenido en formato CSV, tanto en su totalidad como el de los campos y subcampos que se deseen.
- ProCite – Su contenido aparece perfectamente delimitado por comas, aunque carece de cabecera y 20 de los 1.251 registros tienen cadenas de texto que incluyen comillas, lo que impide que se importe de manera adecuada. Para solucionarlo, a través de Notepad++ se añadió una cabecera y se hizo la siguiente búsqueda con expresiones regulares (dejando un espacio delante de las primeras comillas y detrás de las últimas) para localizar y rectificar los problemas de las comillas: "[A-Za-z])|[A-Za-z]"
- Endnote-Refworks – Cada registro aparece en varias líneas, ocupado tantas líneas como campos aparecen en él.
- Pantalla Completa y Presentación Abreviada – Ambos coinciden en ser los que peor recogen la información, junto a Endnote-Refworks. Y es que tiene una estructura visualmente clara, pero no se puede delimitar eficientemente su contenido.

Para lograr obtener la información correctamente de estos tres últimos formatos, se elaboró una aplicación Java (Ilustración 4) por medio de NetBeans IDE 8.0.2. Así, tomando como entrada uno de dichos archivos devuelve un CSV con los datos ya estructurados (ANEXO D).

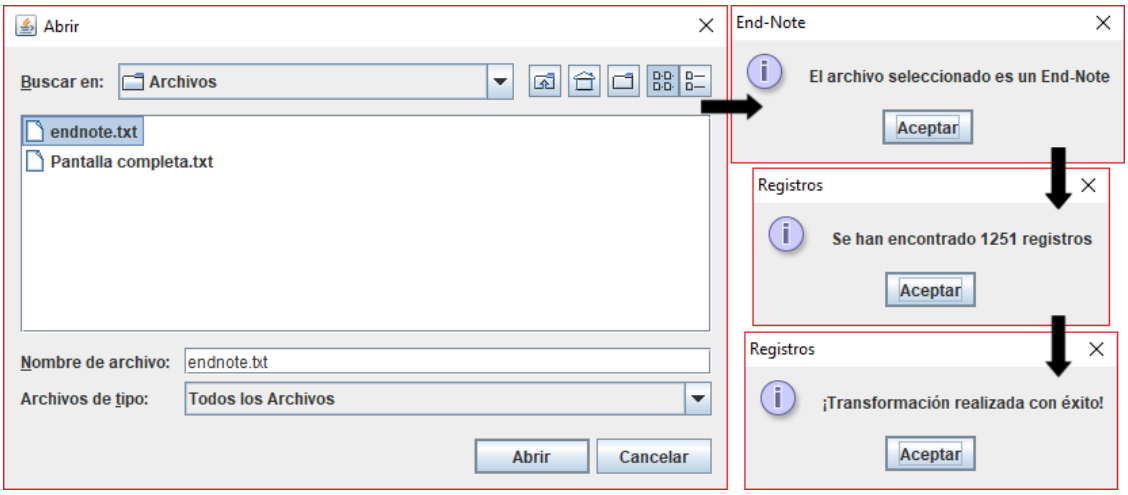

Ilustración 4. Procedimiento para transformar un archivo Endnote-Refworks en un CSV en el programa JAVA

Ya con todos los registros preparados, se llevó a cabo una comparativa de estos (Tabla 10). Así mismo, y más allá de las variaciones, se han detectado dos diferencias llamativas entre los registros en formatos distintos. En primer lugar, los de Pantalla Completa que carecen de autor utilizan la mención de responsabilidad del título (245$c) como tal, y estos junto a ProCite incluyen los resúmenes como notas. Por ello, el formato idóneo para la extracción de los datos es MARC ya que, además de la cantidad, hay un mayor control de los datos.

Antes de proceder a su extracción, también a través de MarcEdit 6, se elaboró un informe en el que se observó la frecuencia de aparición de los diferentes campos y subcampos de MARC 21 (Biblioteca Nacional de España, 2009) para determinar si de los asociados a las necesidades anteriormente descritas había una cantidad mínima para poder realizar el proyecto, así como determinar los 6XX (Campos de encabezamiento de materia) que se usarían. En consecuencia, se estableció el uso del campo 650 (Punto de acceso adicional de materia -Término de materia) por aparecer en el 99 por ciento de ellos (Gráfico 1).

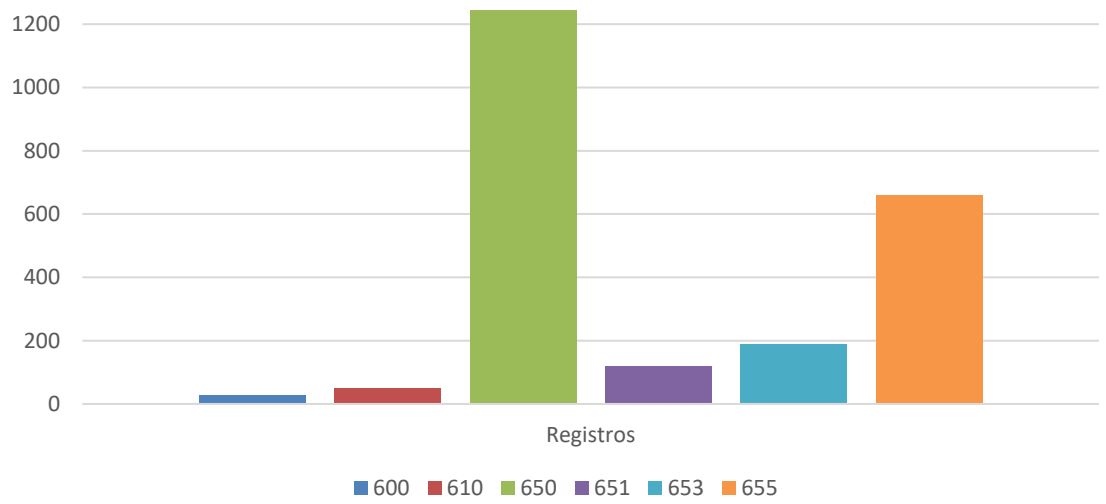

Gráfico 1. Registros del archivo MARC en los que aparecen los diferentes campos de encabezamientos de materia

Junto a los datos que se buscaban en un principio se añadieron además – por tener ahora gracias a este archivo información que puede ser más tarde de mucha utilidad – la cabecera y los campos 001 (Número de control) y 008 (Códigos de información de longitud fija), pero antes hubo que realizar una modificación en el 001 ya que el campo solo estaba presente en 195 registros y no eran valores consecutivos. Desde la herramienta "Añadir/borrar campo" se eliminaron todos los campos 001 para a continuación, desde "Generar números de control", añadirlos en todos los registros de manera secuencial (Ilustración 5).

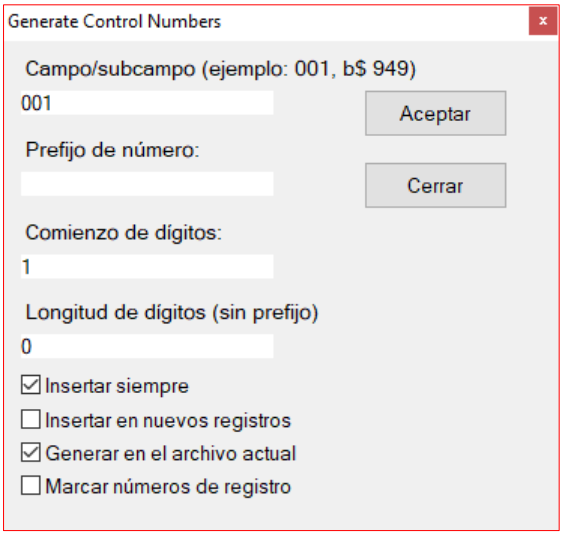

Ilustración 5. Generación de números de control en MarcEdit 6

Finalmente, a través de la opción "Exportar registros delimitados por tabuladores" de MarcEdit 6, se seleccionaron los diferentes campos y subcampos (Tabla 5) para exportarlos en CSV, fijando como delimitador de campos la coma (,), delimitador dentro de campos el punto y coma (;) y delimitador contextual la línea vertical (|).

Información	Campo	Subcampo	Registros
Autor	100 – Punto de acceso principal-Nombre de persona	$a – Nombre de persona	624
		$d – Fechas asociadas al nombre	52
Título	245 – Mención de título	$a – Título	1.251
		$b – Resto del título	583
		$c – Mención de responsabilidad, etc.	653
Publicación	260 – Publicación, distribución, etc.	$a – Lugar de publicación, distribución, etc.	1.243
		$b – Nombre del editor, distribuidor, etc.	1.251
		$c – Fecha de publicación, distribución, etc.	1.183
Materia	650 – Punto de acceso adicional de materia -Término de materia	$a – Término de materia	1.244
		$x – Subdivisión de materia general	195
ISBN	020 – ISBN	$a – ISBN	1.015
Cabecera	Cabecera		1.251
Número	001 – Número de control		1.251
Información	008 – Códigos de información de longitud fija		1.251

Tabla 5. Campos y subcampos de MARC 21 escogidos para su exportación en formato CSV

Por último en esta etapa, se buscó información acerca de la licencia de todos estos datos en la web de la Biblioteca de la Universidad de Granada, pero no se encontró nada al respecto. Es por ello que se tuvo que consultar nuevamente con otro bibliotecario, quien aclaró que estos no se encuentran protegidos bajo ninguna licencia y que su uso es totalmente libre, lo que de tal modo equivale a una licencia de dominio público CC0 1.0 Universal (Ilustración 6).

Debido a que los datos que se van a utilizar se encuentran en dominio público y que la licencia más empleada por proyectos similares es CC0 1.0 Universal, la cual se traduce precisamente en una renuncia a la autoría de los datos, se decidió asignar esta.

Ilustración 6. Sello de la licencia CC0 1.0 Universal

4.4.2.- ETAPA 2: LIMPIAR LOS DATOS

Con los datos necesarios del archivo MARC extraídos en un fichero CSV, se ha escogido una herramienta para su almacenamiento y limpieza (ANEXO C). El software seleccionado ha sido GraphDB 8.0.1 Free, muy similar a OpenRefine pero con posibilidad de lanzar consultas sobre los datos almacenados o externos mediante SPARQL.

Así pues, se importaron los datos (Ilustración 7) a través de la opción “Tabular (OntoRefine)”, la cual también permite cargar archivos MARC pero después no pueden ser modificados de una manera tan rápida y sencilla como con el CSV.

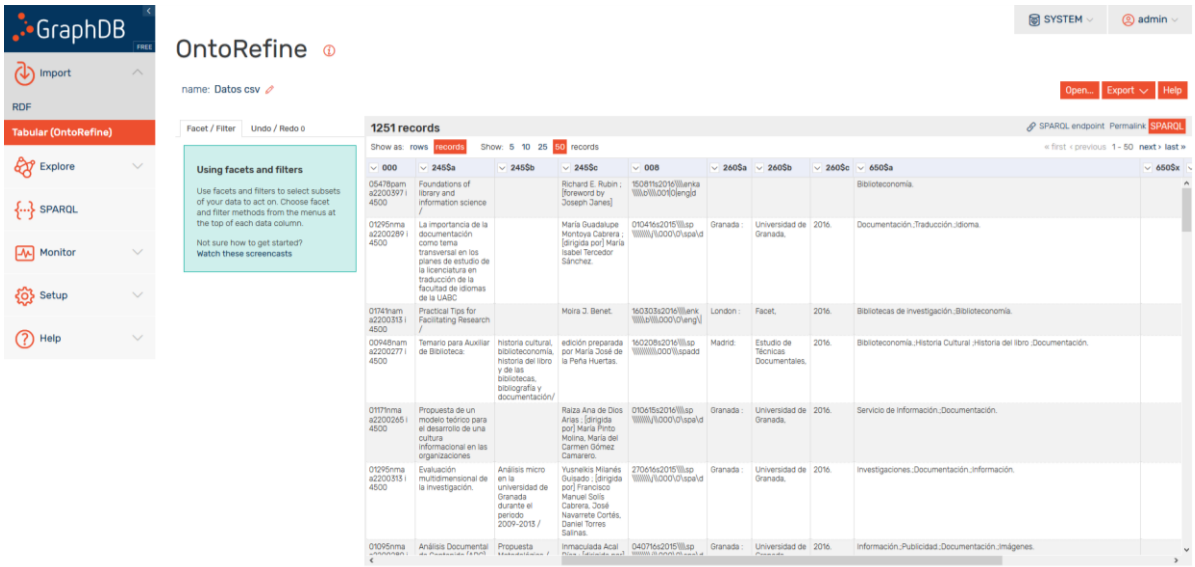

Ilustración 7. Datos cargados en GraphDB

De este modo, uno a uno fueron tratados los datos de cada columna, pertenecientes a cada uno de los campos y subcampos extraídos en la etapa anterior. El objetivo era corregir los posibles errores que tuvieran al máximo posible para obtener "datos limpios y de alta calidad" (Montalvillo Mendizabal, 2012).

Para ello, las columnas se fueron revisando a través de la opción de búsqueda facetada, observando rápidamente la existencia y frecuencia de fallos, como variaciones entre términos que deberían aparecer recogidos bajo una misma faceta pero que no lo hacen.

Una vez localizados, se realizaron las diferentes modificaciones a través de la opción "Transformar", que permite el uso de funciones en GREL (General Refine Expression Language)[15] aplicables a las cadenas de texto de toda las filas de una columna (Morris, 2015), usando además en prácticamente todas las modificaciones expresiones regulares con la sintaxis de JAVA (Oracle, 2016). En definitiva, se ha realizado una modificación de una manera lo más amplia posible debido a la considerable cantidad de datos, en la que la mayoría de los errores se suelen repetir, siendo posible a través de estas expresiones buscar esos patrones entre dichas cadenas de texto y corregirlos, automatizando así el proceso. El desarrollo completo de la limpieza de datos se encuentra detallado en el anexo E.

Junto a los errores, en los que destacan en buena medida los problemas relacionados con los controles de autoridades, puntos de acceso, encabezamientos de materia y ausencia de guiones en el código ISBN (esto último imposibilitará más tarde su conexión con otros códigos que sí los tienen), también se encuentran caracteres especiales o ciertas denominaciones derivadas de la catalogación, como por ejemplo "D.L." en la fecha para indicar que dicho valor se ha extraído del depósito legal, constituyendo estos también una considerable cantidad de modificaciones para dejar en cada celda el dato en bruto.

Hay que destacar que en esta fase se localiza el grueso del proyecto y es la que más esfuerzos y tiempo ha requerido para poder conseguir unos datos tan limpios como ha sido posible. Del éxito de esta etapa se encuentra el poder desarrollar el resto, en especial la de enlazado, donde al cruzar los datos con los de otros datasets es crucial que estos se encuentren en el mejor estado posible, de cara tanto a poder establecer relaciones como garantizar que otros puedan consumirlos de misma manera.

Tras esta etapa se han transformado así los datos procedentes del archivo MARC, solucionando con ello errores en los mismos, otorgándoles una mayor coherencia y utilidad, y extrayendo de un mismo campo o subcampo diferentes datos (Tabla 6).

[15] GREL es un lenguaje para la modificación de valores en celdas similar a las fórmulas de Microsoft Excel.

Además, se llevó también a cabo un proceso similar con datos de la BNE (ANEXO F) a fin de conocer si estos problemas aparecen allí también. En consecuencia, se pudo concluir que a pesar de que no existe ninguna limitación para descargar todo el registro MARC, también en ISO 2709, y que este se encuentra correctamente construido; en cuanto al uso y delimitación de campos y subcampos, siguen existiendo problemas con el control de autoridades, muy en especialmente en el subcampo 260$a, pero no tanto con los puntos de acceso de nombre de persona.

Col.	*Valor original*	*Resultado*
000	00526nam a2200205 i 4500	Nuevo Textual mono
001	1	1
008	090512s1930\\\\uk\\\\\\\\\\\\000\\|\eng\d	eng
100$a	Kenyon, Frederic George,	Kenyon, Frederic George Frederic George Kenyon
100$d	1863-1952.	1863-1952
245$a	LIS education in developing countries :	LIS education in developing countries
245$b	the road ahead /	the road ahead
245$c	edited on behalf of IFLA by Ismail Abdullahi, A.Y. Asundi and C.R. Karisiddappa	edited on behalf of IFLA by Ismail Abdullahi, A.Y. Asundi and C.R. Karisiddappa
260$a	[Jaén] :	Jaén
260$b	[Mundaneum],	Mundaneum
260$c	c2010.	2010
650	\4$aUnión Europea.;\4$aDocumentación$xPublicaciones periódicas.	Unión Europea Documentación—Publicaciones periódicas Documentación Publicaciones periódicas
020$a	1843340534 (paperback);1843340542 (hardback)	1843340534 1843340542

Tabla 6. Ejemplo de transformación de datos

4.4.3.- ETAPA 3: MODELAR

Para la modelización, el primer paso fue el de buscar los vocabularios a emplear, para lo que, además de los vistos durante las fases de análisis anteriores, se emplearon los buscadores vocab.linkeddata.es[16] y Linked Open Data Vocabularies[17]. El vocabulario principal escogido tras ello para convertir los datos obtenidos anteriormente en Linked Data fue BIBFRAME 2.0 (Library of Congress, 2017a), el cual fue publicado en abril de 2016 como segunda fase de dicho proyecto piloto (Library of Congress, 2016), mejorando así algunos problemas vistos en BIBFRAME 1.0. El motivo de esta elección es su carácter internacional, además de que debido a su reciente lanzamiento no hay apenas proyectos que lo hayan implementado y documentado, resultando por todo ello de mayor interés. Para complementarlo se usaron también LC BIBFRAME 2.0 Vocabulary Extension[18] y MADS/RDF Primer[19].

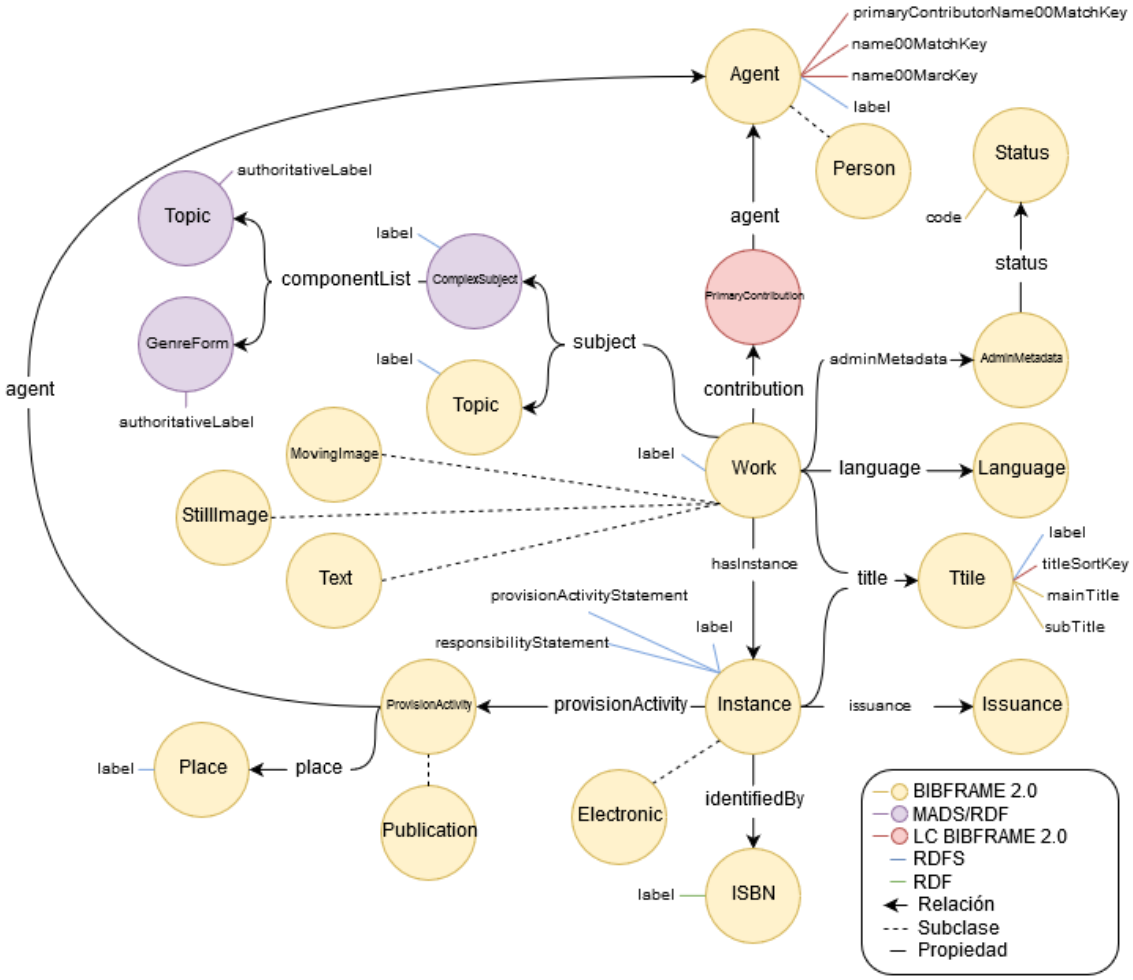

Ilustración 8.Mapa del vocabulario

[16] http://vocab.linkeddata.es/
[17] http://lov.okfn.org/dataset/lov/vocabs
[18] http://id.loc.gov/ontologies/bflc.html
[19] http://www.loc.gov/standards/mads/rdf/

Una vez establecidos los vocabulario a usar, se desarrolló un mapa o red de las diferentes clases y subclases, propiedades y relaciones (Dimou, Heyvaert, Taelman y Verborgh, 2017) que se van a establecer con los datos obtenidos (Ilustración 8), empleando para ello las especificaciones fijadas por la propia Library of Congress para transformar MARC 21 en BIBFRAME 2.0 (Library of Congress, 2017b).

De este modo, los namespaces – delimitadores de los elementos a usar y sus valores esperados – empleados para construir dicha red son los siguientes:

- bf: <http://id.loc.gov/ontologies/bibframe/>
- madsrdf: <http://www.loc.gov/mads/rdf/v1#>
- bflc: <http://id.loc.gov/ontologies/bflc/>
- rdfs: <http://www.w3.org/2000/01/rdf-schema#>
- rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

También se estableció que la URI base que se emplearía luego con los datos durante su modelización sería <http://example.org/>, asignándose por lo tanto las siguientes para estas entidades:

- Work – < http://example.org/numero-de-control#Work>
- Instance – < http://example.org/numero-de-control#Instance>
- Topic – <http://example.org/#Topic650-nombre-de-materia>
- ComplexSubject – <http://example.org/#Topic650-nombre-de-materia>
- Person – < http://example.org/nombre-de-autor>

Aparte de estas URIs también se van a construir otras dos más, combinando dos de los datos disponibles (el código de idioma y el nivel bibliográfico), usando con ello dos schemas de la Library of Congress:

- Language – <http://id.loc.gov/vocabulary/languages/codigo-idioma>[20]
- Issuance – <http://id.loc.gov/vocabulary/issuance/codigo-nivel>[21]

En relación con las URIs, hay que señalar que mientras que con otros vocabularios, como FRBR, para todas las entidades se asignan URIs, en este caso hay clases, como AdminMetada o ProvisionActivity, que no precisan de dicho identificador y que se usan con cierta redundancia para englobar y apuntar a otras clases, complicando en este sentido su elaboración y lectura desde un punto de vista semántico.

[20] http://id.loc.gov/vocabulary/languages.html
[21] http://id.loc.gov/vocabulary/issuance.html

4.4.4.- ETAPA 4: GENERAR

Para la generación, desde MarcEdit 6 existe la posibilidad de convertir un MARCXML en BIBFRAME 2.0 mediante su uso por consola de comandos[22], obteniendo con ello un archivo RDF/XML. Para realizar dicha transformación, el código a ejecutar es el siguiente:

"C:\Program Files\MarcEdit 6\cmarcedit.exe" -s C:\MARC.xml -d C:\BIBFRAME.rdf -utf8 -bibframe bibframe2 -baseuri http://www.example.org/ -output rdfxml -encoding -utf8

No obstante, con ello se estaría pasando por alto la limpieza de datos, manteniéndose así los diferentes errores y consiguientes problemas que ellos traerían. De todos modos, si el archivo MARC careciese de problemas y estuviese listo ya para su modelización esta opción ahorraría mucho trabajo, pero se trata de una posibilidad que no existe ni siquiera, como ya se ha comentado, con los que deberían ser los mejores datos, ya que la propia Biblioteca Nacional de España también precisa de dicha limpieza.

Es por ello que para su elaboración, desde GraphDB, se creó un repositorio que almacenase los datos RDF (Ilustración 9) y por medio del lenguaje SPARQL 1.1 (Harris y Seaborne, 2013), también a través de dicha plataforma, se fueron elaborando y ejecutando las diferentes consultas para crear los grafos con BIBFRAME 2.0, todas ellas detalladas en el anexo G.

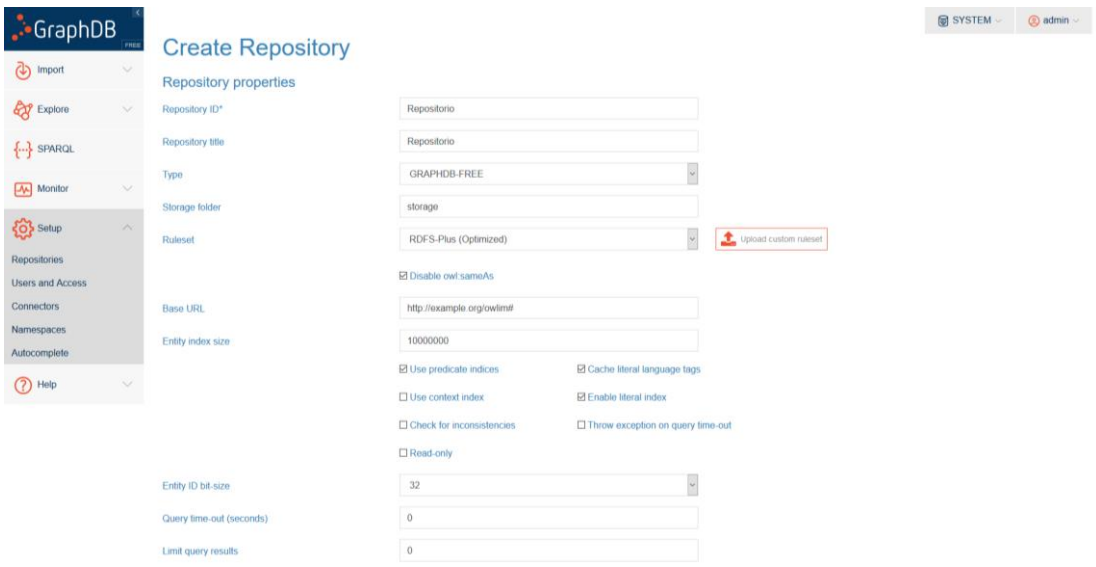

Ilustración 9. Creación de un repositorio en GraphDB

De este modo, se ejecutaron las diferentes consultas en SPARQL, las cuales recogen los datos anteriormente limpiados del CSV, buscando por filas y columnas, y los transforma en grafos, siendo esta la etapa más compleja, aunque ligeramente más corta que la limpieza de datos. El

[22] http://marcedit.reeset.net/cmarcedit-exe-using-the-command-line

principal problema encontrado en este proceso está en los campos vacíos y el uso del operador 'OPTIONAL' para indicar dicha carencia de valores en algunos de sus registros o filas, para evitar que de modo contrario dicha consulta acabe restringiéndose únicamente a los registros con valores en todos sus campos. Sin embargo, fueron necesarias varias consultas ya que, al querer abarcar tanto, en algunas clases se generaban grafos vacíos. Para superar este proceso hubo incluso que recurrir a la comunidad de programadores Stack Overflow en dos ocasiones[23][24], pero finalmente se obtuvieron las marcadas 22 clases y subclases (Ilustración 10), coincidiendo el contenido de cada una con las instancias previstas.

Ilustración 10. Clases y subclases obtenidas y número de instancias

[23] https://stackoverflow.com/questions/42760251/rdfising-data-with-sparql-and-spin
[24] https://stackoverflow.com/questions/43926092/sparql-construct-insert-query-and-blank-nodes

Ya con los grafos construidos, el siguiente y último paso que se realizó en esta etapa fue el de su validación, lo cual se hizo a través de IDLab Turtle Validator[25]. Desde GraphDB se llevó a cabo una exportación de todos los grafos almacenados, marcando como serialización Turtle ya que es el formato exigido para hacerlo. Tras ello, se copió el contenido de ese archivo, tanto los namespaces como las tripletas, se pegaron en el validador y este se ejecutó con éxito (Ilustración 11).

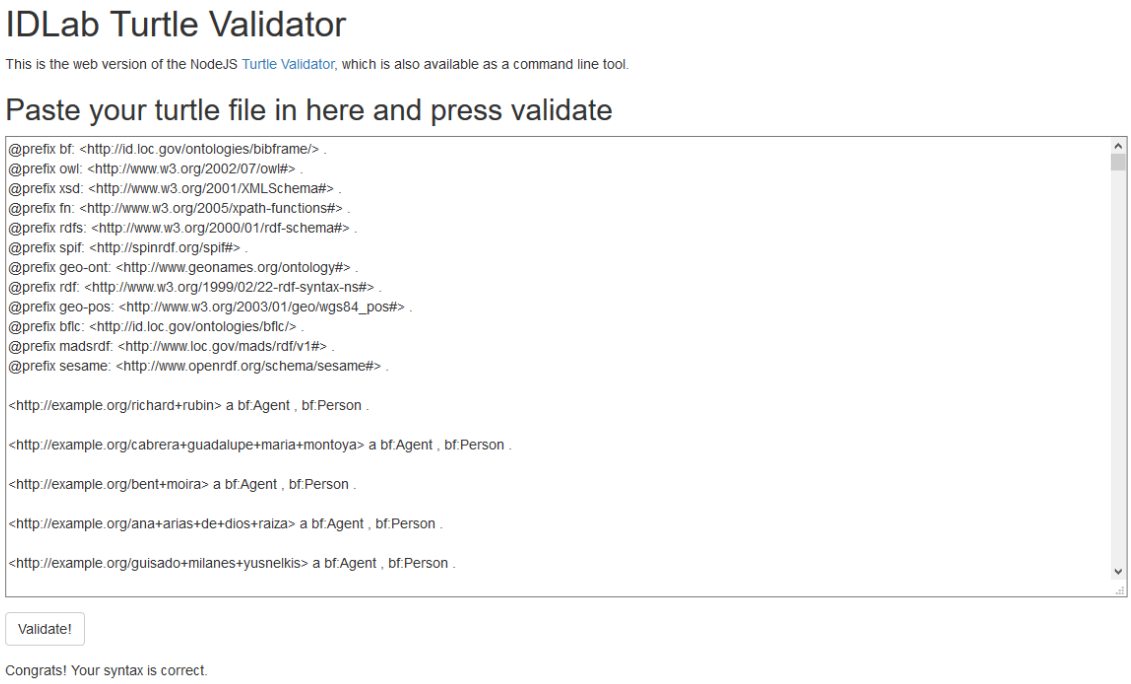

Ilustración 11. Validación del dataset en IDLab Turtle Validator

4.4.5.- ETAPA 5: ENLAZAR

En este apartado se buscaron los diferentes y potenciales datasets para el establecimiento de relaciones con el conjunto de datos antes elaborado, también a través de GraphDB.

No obstante, antes de llevar a cabo este proceso es necesario tener en cuenta que por limitaciones de la licencia gratuita de esta plataforma no es posible cargar grandes dumps files o volcados de datos en RDF, como por ejemplo el de VIAF, por lo que la prioridad será la de buscar aquellos que permitan acceder a sus tripletas a través de un SPARQL Endpoint.

Para localizar esas colecciones de datos con las que poder buscar relaciones y llevar a cabo el enriquecimiento de los datos elaborados, y además con dicha forma de acceso descrita, se usó Datahub, buscando datasets relacionados con bibliotecas y registros de autoridad. De entre los

[25] http://ttl.summerofcode.be/

resultados obtenidos, finalmente se optaron por dos conjuntos de datos que satisfacían dichas condiciones y se encontraban actualizados y activos:

- Lista de Encabezamientos de materia para las Bibliotecas Públicas (LEMB) – http://id.sgcb.mcu.es/sparql
- Biblioteca Nacional de España (BNE) – http://datos.bne.es/sparql

Antes de llevar a cabo los mismos pasos que en la etapa anterior, se ejecutaron varias consultas desde el SPARQL de GraphDB, sin insertar datos en el repositorio, cruzando los diferentes datos con el objetivo de encontrar significativas y numerosas relaciones entre ambos conjuntos.

De este modo, con la Lista de Encabezamientos de materia para las Bibliotecas Públicas se localizaron rápidamente vínculos entre la clase Topic de BIBFRAME y Concept de SKOS, usada por LEMB, concretamente para 211 de los 473 encabezamientos de materias almacenados en el repositorio. Destacar en esta parte que otro de los aspectos que hay que cuidar, además del valor del dato en sí, es el del datatype o tipo de dato, pues en este caso por ejemplo, además de especificarse que se trata de un cadena de texto también se indica en el dataset de LEMB el idioma, en este caso el español. Por ello, si ambos no coinciden al completo la búsqueda no obtendrá ningún resultado.

Por su parte, la BNE planteó más problemas, y es que en este caso, con una colección mucho más amplia, en la que se pueden cruzar una mayor cantidad y variedad de datos, siendo en este caso el ISBN el más atractivo para tales fines por poder conectar dos instancias de manera inequívoca, hubo dos problemas. Por un lado, los códigos obtenidos de la Biblioteca de la Universidad de Granada aparecen sin ningún guion, mientras que los de la Biblioteca Nacional de España sí los incluye. Se probaron diversas alternativas a ello, pero entonces se encontró con ello el segundo y más importante problema, y es que, al menos durante el periodo en el que se estuvieron realizando dichas pruebas, no siempre se recuperaban todos los datos de las instancias de la BNE, entre ellos el ISBN, e incluso a veces era imposible recuperar alguno.

Por este motivo, se volvió a revisar de nuevo los datos ofrecidos por la BNE, más allá del SPARQL Endpoint, y se encontró un dump file que establecía una equivalencia entre las URIs de la clase autores de la BNE y su registro en VIAF[26]. Este pudo ser importado a GraphDB, su tamaño no lo impedía, y a través de dos consultas, una para insertarle a cada grafo del

[26] https://datahub.io/dataset/datos-bne-es/resource/bb29e8ff-5f39-418f-b049-689479ac440a

dump file el nombre completo del autor y otra para finalmente poder buscar entre ellos alguno de los 452 nombres de autores y asociarles su VIAF, se consiguieron encontrar para 184 de ellos.

Una vez localizados los datos que se iban a relacionar, se repitió el proceso anterior: se elaboró un mapa del nuevo dataset (Ilustración 12), se llevaron a cabo las diferentes consultas SPARQL para insertar dichos datos (ANEXO H) y se validó con éxito de la misma forma que antes este conjunto de datos.

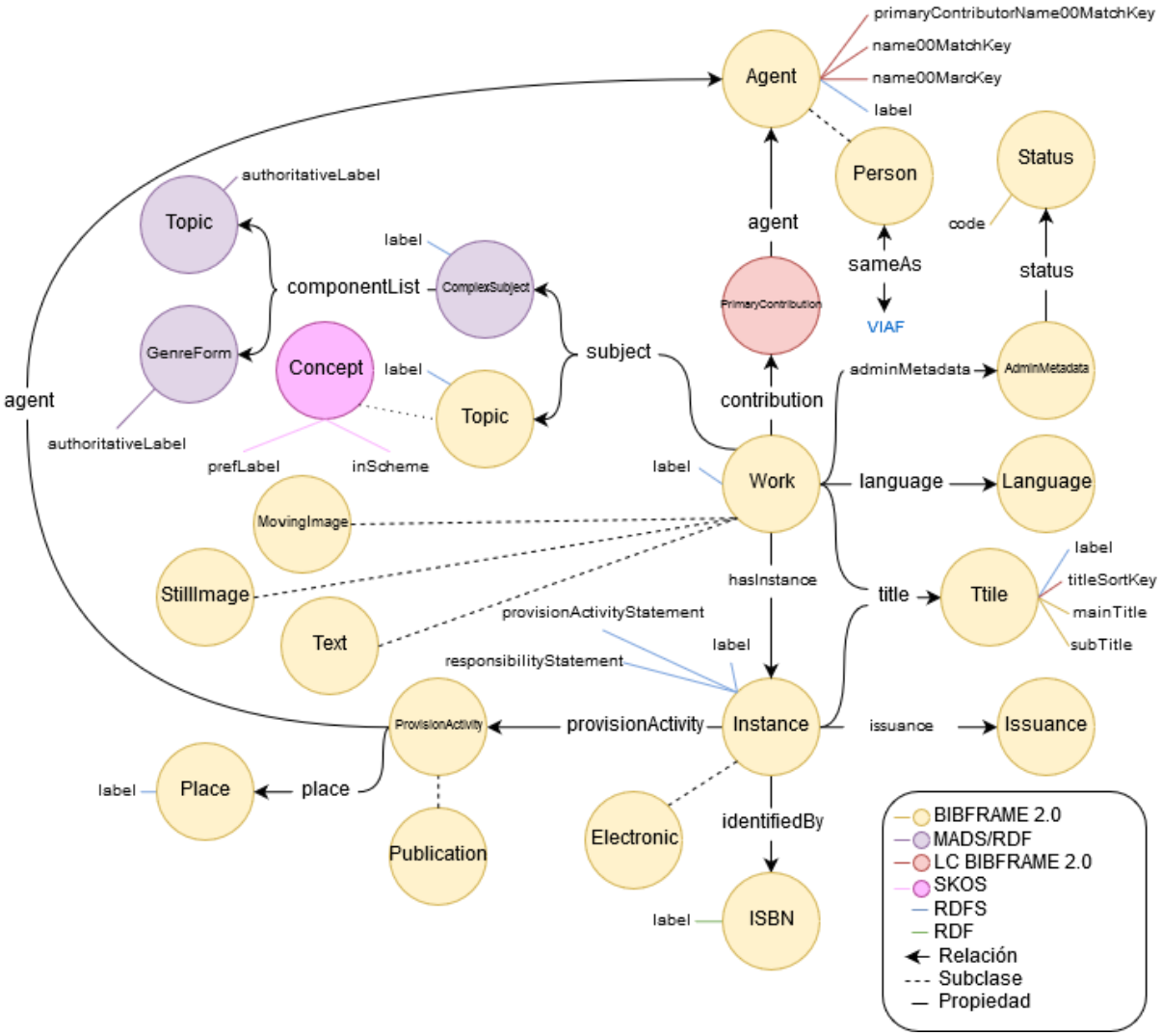

Ilustración 12. Mapa del vocabulario tras su enriquecimiento con datos de la LEMB y BNE

4.4.6.- ETAPA 6: PUBLICAR

Por último, se ha publicado el dataset en Internet. El primer paso para ello es el de determinar el método de acceso y tecnologías para su publicación. De este modo, se decidió publicarlo a

través de una página web, del mismo que la Biblioteca Nacional de España, un SPARQL Endpoint y un dump file.

Para su publicación en un sitio web se pensó en un primer momento en llevarlo a cabo a través de Drupal, no obstante la falta de plugins, actualización y documentación de los mismos para llevar a cabo tal proceso hizo imposible tomar dicha vía. Se revisaron otros CMS usados por otros proyectos, como Joomla!, pero se obtuvo idéntico resultado.

Es por ello que finalmente se optó por crear un sitio Wordpress (http://wenceslaoarroyomachado.es/) en el que publicar el dump file, acompañado de sus metadatos, así como del resto de materiales producidos durante todo el proceso para que todos los que quieran puedan servirse de ellos. Además, se probaron varias de las herramientas más usadas por proyectos Linked Data de bibliotecas, archivos y museos (Smith-Yoshimura, 2016) buscando crear un SPARQL Endpoint. Para ello, usando mi portátil como servidor local, por medio de Ubuntu 14.04.5 LTS, se han instalado las siguientes plataformas con tal objetivo:

- Openlink Virtuoso 7.2.4[27] – http://wenceslao.technology:8890/
- Blazegraph 2.1.4[28] – http://wenceslao.technology:9999/
- GraphDB 8.0.1 Free – http://wenceslao.technology:7200/

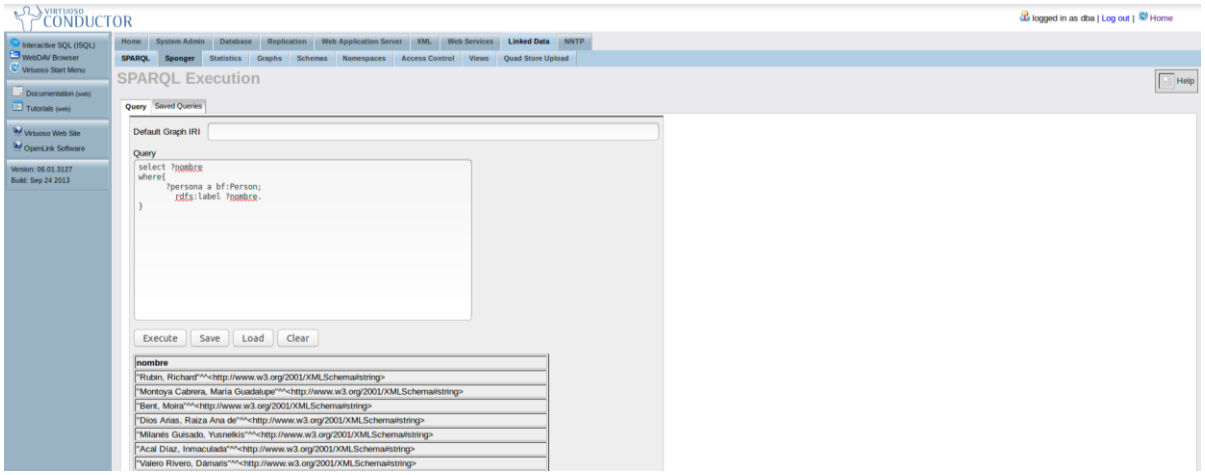

Ilustración 13. Búsqueda de nombres de autores en el SPARQL Endpoint de Openlink Virtuoso

En las tres plataformas se ha creado un usuario para poder usar SPARQL Endpoint (Ilustración 13) y otras de las posibilidades que ofrecen, como la exploración visual de grafos (Ilustración 14), siendo el nombre y contraseña para ello en todas "usuario".

[27] https://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/VOSDownload
[28] https://www.blazegraph.com/download/

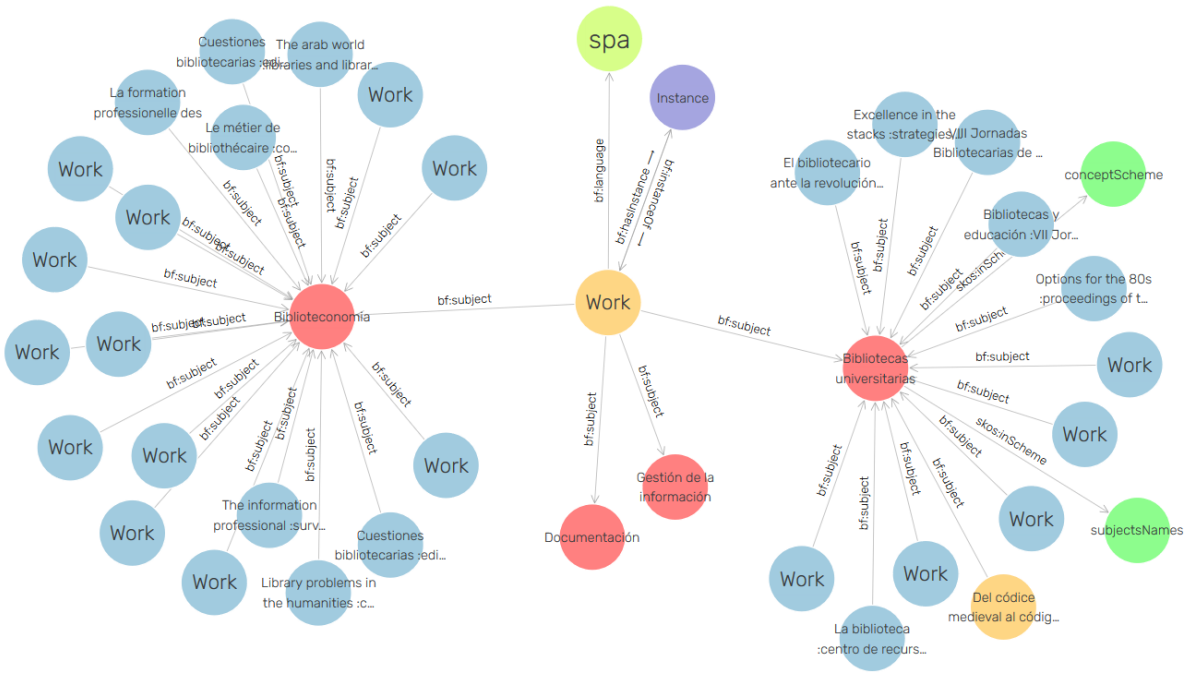

Ilustración 14. Exploración visual de grafos en GraphDB

Por último, es necesario elaborar unos metadatos que acompañen al dataset, un elemento importante para poder rastrear el origen de los datos y hacerlos más accesibles (Heath y Bizer, 2011). Debido a ello se han creado unos metadatos con un campos mínimos (Center for Government Excellence, 2016) con Dublin Core Metadata Element Set[29] (Tabla 7), insertándolos en el dataset en sí mismo, por medio de tripletas, como en el Wordpress.

Campo	*Valor*
dc:creator	Arroyo Machado, Wenceslao
dc:title	Registros bibliográficos de documentación y biblioteconomía
dc:description	Registros bibliográficos extraídos de la Biblioteca de la Universidad de Granada sobre documentación y biblioteconomía
dc:type	Dataset
dc:subject	Documentación
dc:license	CC0 1.0 Universal
dc:date	14 de junio de 2017

Tabla 7. Metadatos creados para el dataset

[29] http://dublincore.org/documents/2012/06/14/dcmi-terms/

5.- CONCLUSIONES

- Se ha logrado diseñar una metodología para transformar registros bibliográficos en Linked Data e implementarla alcanzando las cinco estrellas del Linked Open Data.
- No existe una única metodología, vocabulario u ontología, formato de serialización, ni tecnología para la publicación de Linked Data, echándose en falta al respecto un CMS que explote y facilite el acceso y presentación de datos en RDF.
- La extracción de los datos de su fuente, su limpieza y generación de RDF, esta última tras llevar a cabo la modelización, son procesos que una vez realizados y analizados pueden ser completamente automatizados de cara a la actualización e incorporación de nuevos datos.
- La etapa de limpieza de datos es el grueso de la implementación y parte fundamental para el éxito del proyecto, y en donde se ha puesto de manifiesto la importancia y necesidad de que la Biblioteca de la Universidad de Granada realicé una profunda revisión de sus registros y métodos de acceso y filtrado, en especial en lo referido al control de autoridades y puntos de acceso, en donde la Biblioteca Nacional de España peca de lo mismo, así como del correcto uso de los códigos en las cabeceras en MARC 21, la descripción de materias y formato de códigos ISBN.
- La posibilidad y facilidad para elaborar un vocabulario u ontología propia favorece la resolución de los problemas y objetivos particulares de cada proyecto, pero puede dificultar su interoperabilidad, precisándose una normalización en dicho sentido.
- BIBFRAME, vocabulario en continua evolución, dista a día de hoy de ser un modelo idóneo, siendo su uso muy reducido y generando mucha redundancia, aspecto que afecta tanto a su elaboración como lectura.
- Para la conversión de los registros bibliográficos a Linked Data, su enriquecimiento y publicación; el personal, recursos y requerimientos necesarios para ello no suponen una auténtica barrera, existiendo un amplio abanico de alternativas y herramientas multiplataforma y de código abierto, por lo que el principal obstáculo percibido son los esfuerzos iniciales necesarios, requiriéndose en este sentido conocimientos en lenguajes de programación, consulta de bases de datos y expresiones regulares.
- El enriquecimiento de los datos es uno de los principales y más visibles atractivos a la hora de dar el paso al Linked Data, dependiendo de su éxito la calidad de los datos con los que se trabaja.

BIBLIOGRAFÍA

Adida, B., Birbeck, M., McCarron, S. y Herman, I., 2015. *RDFa Core 1.1 - Third Edition*. [en línea] Disponible en: <https://www.w3.org/TR/rdfa-syntax/> [Accedido 2 mar. 2017].

AKSW, 2016. *Projects*. [en línea] Disponible en: <http://aksw.org/Projects.html> [Accedido 5 feb. 2017].

Beckett, D., 2014. *RDF 1.1 N-Triples*. [en línea] Disponible en: <https://www.w3.org/TR/n-triples/> [Accedido 2 mar. 2017].

Beckett, D., Berners-Lee, T., Prud'hommeaux, E. y Carothers, G., 2014. *RDF 1.1 Turtle*. [en línea] Disponible en: <https://www.w3.org/TR/turtle/> [Accedido 2 mar. 2017].

Bermès, E., Coyle, K. y Dunsire, G., 2011. *Library Linked Data Incubator Group Final Report*. [en línea] Disponible en: <https://www.w3.org/2005/Incubator/lld/XGR-lld-20111025/> [Accedido 9 ene. 2017].

Berners-Lee, T., 2006. *Linked Data - Design Issues*. [en línea] Disponible en: <https://www.w3.org/DesignIssues/LinkedData.html> [Accedido 2 ene. 2017].

Biblioteca del Congreso Nacional de Chile, 2012. *Linked Open Data: ¿Qué es? — Sitio Datos abiertos enlazados*. [en línea] Disponible en: <http://datos.bcn.cl/es/informacion/que-es> [Accedido 3 ene. 2017].

Biblioteca Nacional de España, 2009. *MARC 21 para Registros Bibliográficos*. [en línea] Disponible en: <http://www.bne.es/es/Micrositios/Guias/Marc21/> [Accedido 21 ene. 2017].

Bustán, G., María, A. y González, D.F.J., 2016. *Principios y tecnologías linked data para la publicación de datos académicos de las diferentes carreras de la Universidad Nacional de Loja.* [en línea] Disponible en: <http://dspace.unl.edu.ec:9001//handle/123456789/11202>.

Center for Government Excellence, 2016. *Open Data - Metadata Guide*. [en línea] GitBook. Disponible en: <https://www.gitbook.com/book/centerforgov/open-data-metadata-guide/details> [Accedido 25 may 2017].

Cisco, 2017. *The Zettabyte Era—Trends and Analysis*. [en línea] Disponible en: <http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html> [Accedido 9 jun. 2017].

Deliot, C., 2014. Publishing the British National Bibliography as Linked Open Data. *The British Library*. [en línea] Disponible en: <http://microblogging.infodocs.eu/wp-content/uploads/2014/10/publishing_bnb_as_lod.pdf>.

Dimou, A., Heyvaert, P., Taelman, R. y Verborgh, R., 2017. Modeling, Generating, and Publishing Knowledge as Linked Data. [en línea] Springer, Cham, pp.3-14. Disponible en:

<http://link.springer.com/10.1007/978-3-319-58694-6_1> [Accedido 12 jun. 2017].
Europeana, 2015. *Data structure*. [en línea] Disponible en: <http://labs.europeana.eu/api/linked-open-data-data-structure> [Accedido 3 feb. 2017].
García Lorca, F., 1996. *Alocución al pueblo de Fuente Vaqueros*. Diputación Provincial de Granada, Patronato Cultural Federico García Lorca.
Gómez Ruiz, A., 2013. Aplicación bibliográfica usando Linked Data. [en línea] Disponible en: <https://eciencia.urjc.es/handle/10115/12103>.
Hallo, M., Luján-Mora, S., Maté, A. y Trujillo, J., 2016. Current state of Linked Data in digital libraries. *Journal of Information Science*, [en línea] 42(2), pp.117-127. Disponible en: <http://jis.sagepub.com/content/42/2/117>.
Hallo, M., Lujan-Mora, S. y Trujillo, J., 2014. Transforming Library Catalogs into Linked Data. *Iceri2014: 7th International Conference of Education, Research and Innovation*, pp.1845-1853.
Harris, S. y Seaborne, A., 2013. *SPARQL 1.1 Query Language*. [en línea] Disponible en: <https://www.w3.org/TR/2013/REC-sparql11-query-20130321/> [Accedido 4 may 2017].
Hastings, R., 2015. Linked Data in Libraries: Status and Future Direction. *Computers in Libraries*. [en línea] Disponible en: <http://www.infotoday.com/cilmag/nov15/Hastings--Linked-Data-in-Libraries.shtml>.
Heath, T. y Bizer, C., 2011. Linked Data: Evolving the Web into a Global Data Space. *Synthesis Lectures on the Semantic Web: Theory and Technology*, [en línea] 1(1), pp.1-136. Disponible en: <http://www.morganclaypool.com/doi/abs/10.2200/s00334ed1v01y201102wbe001>.
Hyland, B., Atemezing, G. y Villazón-Terrazas, B., 2014. *Best Practices for Publishing Linked Data*. [en línea] Disponible en: <https://www.w3.org/TR/ld-bp/> [Accedido 8 abr. 2017].
Hyvönen, E., Tuominen, J., Alonen, M. y Mäkelä, E., 2014. Linked Data Finland: A 7-star Model and Platform for Publishing and Re-using Linked Datasets. [en línea] Springer, Cham, pp.226-230. Disponible en: <http://link.springer.com/10.1007/978-3-319-11955-7_24> [Accedido 14 jun. 2017].
IFLA, 2014. *Archival Resource Key (ARK)*. [en línea] Disponible en: <http://www.ifla.org/best-practice-for-national-bibliographic-agencies-in-a-digital-age/node/8793> [Accedido 9 ene. 2017].
Import.io, 2015. *All the best big data tools and how to use them*. [en línea] Disponible en:

<https://www.import.io/post/all-the-best-big-data-tools-and-how-to-use-them/> [Accedido 5 feb. 2017].

Internet Live Stats, 2017. *Total number of Websites*. [en línea] Disponible en: <http://www.internetlivestats.com/total-number-of-websites/> [Accedido 9 jun. 2017].

Library of Congress, 2016. *BIBFRAME Pilot (Phase One—Sept. 8, 2015 – March 31, 2016): Report and Assessment*. [en línea] Disponible en: <http://www.loc.gov/bibframe/docs/pdf/bibframe-pilot-phase1-analysis.pdf>.

Library of Congress, 2017a. *BIBFRAME - Bibliographic Framework Initiative (Library of Congress)*. [en línea] Disponible en: <http://www.loc.gov/bibframe/> [Accedido 4 mar. 2017].

Library of Congress, 2017b. *MARC 21 to BIBFRAME 2.0 Conversion Specifications (BIBFRAME - Bibliographic Framework Initiative, Library of Congress)*. [en línea] Disponible en: <https://www.loc.gov/bibframe/mtbf/> [Accedido 4 mar. 2017].

Lóscio, B.F., Burle, C. y Calegari, N., 2016. *Data on the Web Best Practices*. [en línea] Disponible en: <https://www.w3.org/TR/dwbp/> [Accedido 9 ene. 2017].

MacKenzie., S., Carl, G., Stahmer, X.L. y Gloria, G., 2017. *BIBFLOW: A Roadmap for Library Linked Data Transition*. [en línea] Disponible en: <https://bibflow.library.ucdavis.edu/wp-content/uploads/2017/03/bibflow_roadmap_revised_3_14_2017.pdf>.

Manola, F., Miller, E. y McBride, B., 2014. *RDF 1.1 Primer*. [en línea] Disponible en: <https://www.w3.org/TR/2014/NOTE-rdf11-primer-20140225/> [Accedido 3 ene. 2017].

Montalvillo Mendizabal, L., 2012. Definición y desarrollo de herramienta Web de gestión de metadatos Business Intelligence. pp.17-18.

Morris, T., 2015. *General Refine Expression Language*. [en línea] Disponible en: <https://github.com/OpenRefine/OpenRefine/wiki/General-Refine-Expression-Language> [Accedido 20 feb. 2017].

MyUtilsJava, 2015. *Generar y Leer CSV desde Java*. [en línea] Disponible en: <http://www.myutilsjava.net/tutoriales/index.php/java/49-generar-y-leer-csv-desde-java> [Accedido 5 ene. 2017].

Norma UNE-ISO 2709:2006. http://www.aenor.es/. [en línea] Disponible en: <http://www.aenor.es/aenor/normas/normas/fichanorma.asp?tipo=N&codigo=N0036928#.WUJo-3lLeUk>.

Oracle, 2016. *Pattern*. [en línea] Disponible en: <https://docs.oracle.com/javase/7/docs/api/java/util/regex/Pattern.html> [Accedido 26 ene.

2017].
Papadakis, I., Kyprianos, K. y Stefanidakis, M., 2015. Linked Data URIs and Libraries: The Story So Far. *The Magazine of Digital Library Research*, 21.
Sánchez, J.A.P., 2016. *VIII Encuentros de Centros de Documentación de Arte Contemporáneo*. Disponible en: <https://issuu.com/artium_vitoria/docs/artium-2016?reader3=1>.
Smith-Yoshimura, K., 2016. Analysis of International Linked Data Survey for Implementers. *D-Lib Magazine*, [en línea] Volume 22,. Disponible en: <http://www.dlib.org/dlib/july16/smith-yoshimura/07smith-yoshimura.html>.
Sporny, M., Longley, D., Kellogg, G., Lanthaler, M. y Niklas, L., 2014. *JSON-LD 1.0*. [en línea] Disponible en: <https://www.w3.org/TR/json-ld/> [Accedido 3 ene. 2017].
Sulé, A., Centelles, M., Franganillo, J. y Gascón, J., 2016. Aplicación del modelo de datos RDF en las colecciones digitales de bibliotecas, archivos y museos de España. *Revista española de Documentación Científica*, [en línea] 39(1), p.e121. Disponible en: <http://redc.revistas.csic.es/index.php/redc/article/view/924>.
Taylor, S., Jekjantuk, N., Mellish, C. y Pan, J.Z., 2014. Reasoning Driven Configuration of Linked Data Content Management Systems. *Semantic Technology*, 8388, pp.429-444.
Torre-Bastida, A.-I., González-Rodríguez, M. y Villar-Rodríguez, E., 2015. Datos abiertos enlazados (LOD) y su implantación en bibliotecas: iniciativas y tecnologías. *El Profesional de la Información*, [en línea] 24(2), pp.113-120. Disponible en: <http://recyt.fecyt.es/index.php/EPI/article/view/epi.2015.mar.04>.
Vila-Suero, D., Villazon-Terrazas, B. y Gomez-Perez, A., 2013. datos.bne.es: A library linked dataset. *Semantic Web*, 4(3), pp.307-313.
W3C, 2011. *TaskForces/CommunityProjects/LinkingOpenData/DataLicensing - W3C Wiki*. [en línea] Disponible en: <https://www.w3.org/wiki/TaskForces/CommunityProjects/LinkingOpenData/DataLicensing> [Accedido 12 ene. 2017].
W3C, 2017. *Guía Breve de Linked Data*. [en línea] Disponible en: <http://www.w3c.es/Divulgacion/GuiasBreves/LinkedData> [Accedido 3 ene. 2017].
Wenz, R., 2013. Linked open data for new library services: the example of data.bnf.fr. *JLIS.it*, [en línea] 4(1), p.403. Disponible en: <http://search.proquest.com/docview/1270767856>.
Wikipedia, 2017. *EndNote*. [en línea] Disponible en: <https://en.wikipedia.org/wiki/EndNote> [Accedido 6 ene. 2017].

ANEXO

A. LINKED DATA EN LA CIENCIA

Desde el nacimiento del Linked Data hasta ahora, las publicaciones en revistas científicas, referidas tanto al Linked Data como a Linked Open Data, no han dejado de crecer, llegando en 2016 a las 700 publicaciones en la base de datos Scopus, y a las 667 en Web of Science (Ilustración 15). Siendo 2015 el mejor año para ambas con 780 y 699 publicaciones, respectivamente.

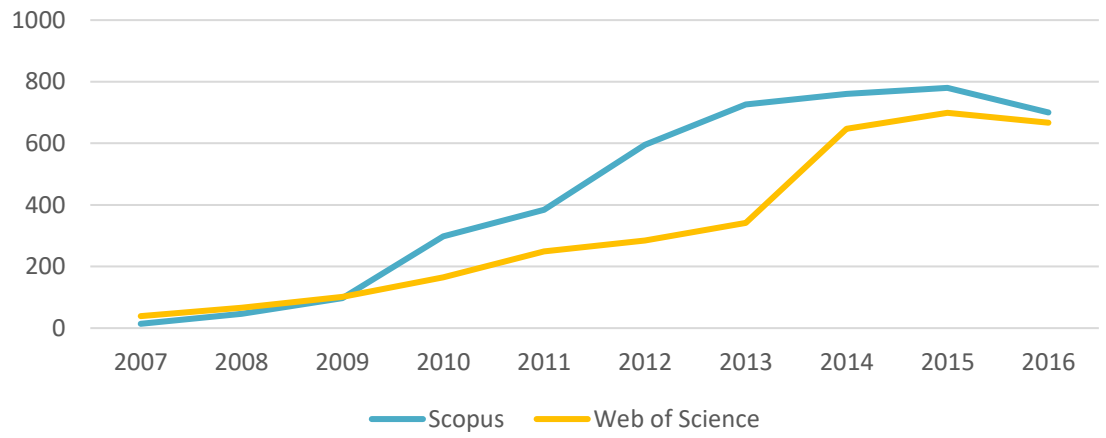

Ilustración 15. Publicaciones sobre Linked Data o Linked Open Data publicadas en Scopus y Web of Science entre 2007 y 2016

Del mismo modo, también es posible apreciar un crecimiento y punto máximo paralelo de dichas publicaciones en el ámbito de las bibliotecas (Ilustración 16).

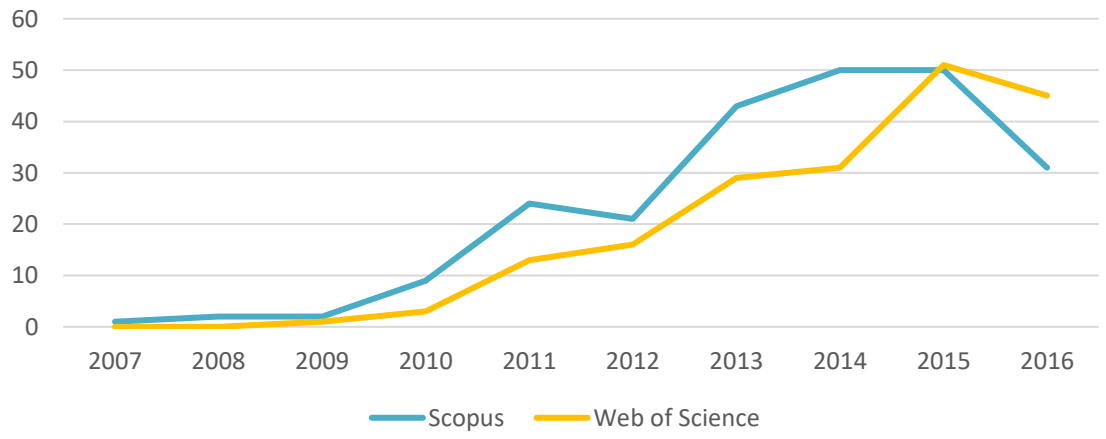

Ilustración 16. Publicaciones sobre Linked Data y bibliotecas publicadas en Scopus y Web of Science entre 2007 y 2016

A través de Scopus, ya que se puede observar en las anteriores gráficas una clara superioridad en el número de trabajos, se buscaron todas las publicaciones de revistas científicas relacionados con el Linked Data o Linked Open Data, no solo las ubicados entre 2007 y 2016, conformando un total de 4.606 en el momento en el que se realizó la búsqueda, y fueron descargados en formato CSV incluyendo toda la información posible para crear diferentes mapas por medio del software VOSviewer (versión 1.6.4)[30].

En primer lugar, se creó un mapa basado en la información bibliográfica de dichos registros descargados. Se realizó un análisis de coocurrencias, usando como unidad de análisis las palabras clave que figuran en dichos documentos y que han sido establecidas por los propios autores, por ser las más representativas del contenido, frente a las que establece Scopus, que mejor representan la disciplina. Para el recuento de esas coocurrencias se usó el recuento fraccionado (fractional counting) frente al completo (full counting), para no aumentar el peso de algunos términos de manera artificial. El número mínimo de coocurrencias que tienen que existir para que aparezcan representadas se fijó en 20, de cara a reducir el número de términos y seleccionar aquellos más relevantes.

Entre los 59 términos representados en el mapa, existían dos parejas de términos (lod - linked open data y ontology - ontolgies) que representaban lo mismo, por ello se fusionaron a través de un tesauro de términos, quedando únicamente linked open data y ontologies.

El resultado de este mapa (Ilustración 17) es muy llamativo, ya que el primer cluster que se forma (color rojo) incluye los términos libraries y digital libraries junto a semantic web, ontologies, metadata y linked open data, entre otros. Tanto libraries como digital libraries están directamente vinculados a linked data, el cual es el nexo común entre todos ellos y núcleo del mapa, pero el segundo también lo está con el término metadata. Todo esto refleja la importancia y papel de las bibliotecas en lo que a Linked Data se refiere en el mundo de las publicaciones.

[30] http://www.vosviewer.com/

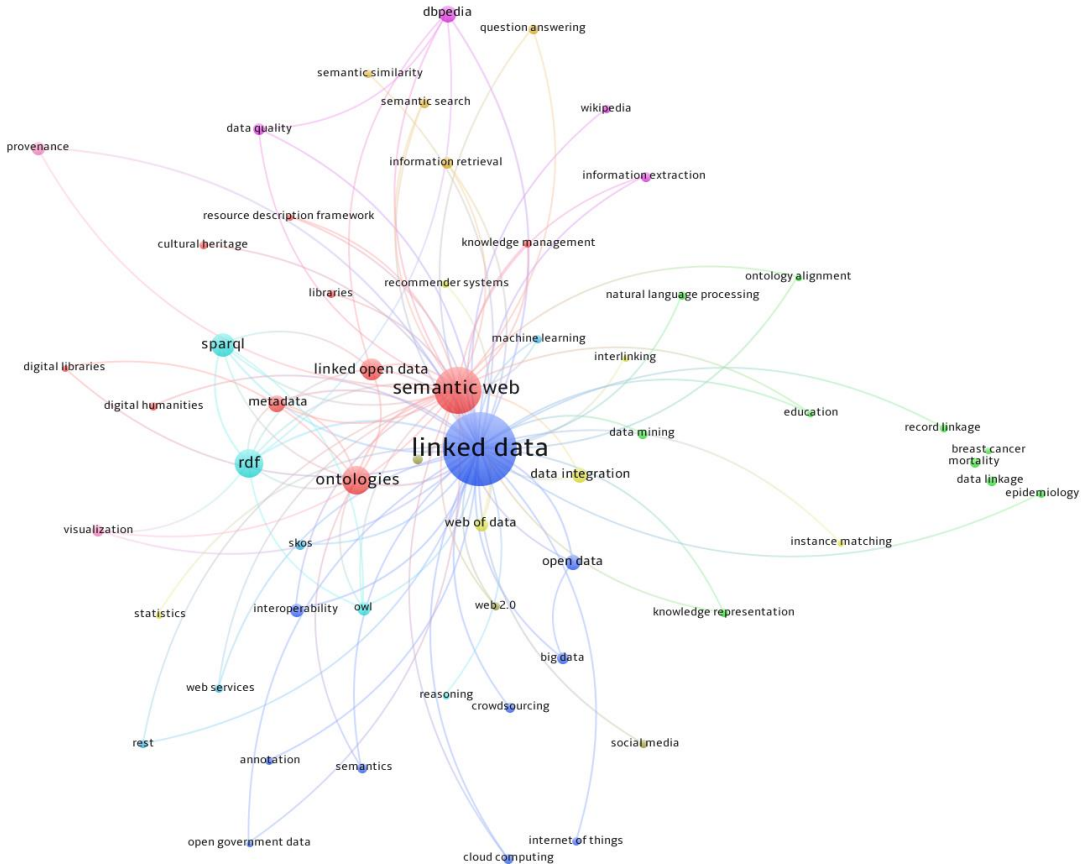

Ilustración 17. Mapa de coocurrencias de palabras clave de los autores en las publicaciones sobre Linked Data o Linked Open Data en Scopus

Por otro lado, se realizó otro mapa basado en la misma información bibliográfica, pero realizando un análisis de coautorías, en el que la unidad de análisis fueron los países y el método de recuento el fraccional, por el mismo motivo que antes. El número mínimo de coocurrencias se fijó en 57 trabajos mínimos entre dos o más países, para obtener los 20 países que más colaboran y observar mejor las relaciones entre ellos.

El mapa (Ilustración 18) muestra que los países con más colaboraciones son Estados Unidos, Alemania, Reino Unido y España, sin embargo, se encuentran casi todos en clusters diferentes, lo que implica que dentro de esos grupos son aquellos países con los que todos quieren colaborar. Mientras que Estados Unidos y Reino unido van muy de la mano junto a Canada, Australia, China y Japón, España hace lo propio con Bélgica, Grecia, Austria, Holanda y Suiza. Por su parte Alemania mantiene buena parte de sus colaboraciones con Francia, Irlanda, Italia, Corea del Sur y Brasil.

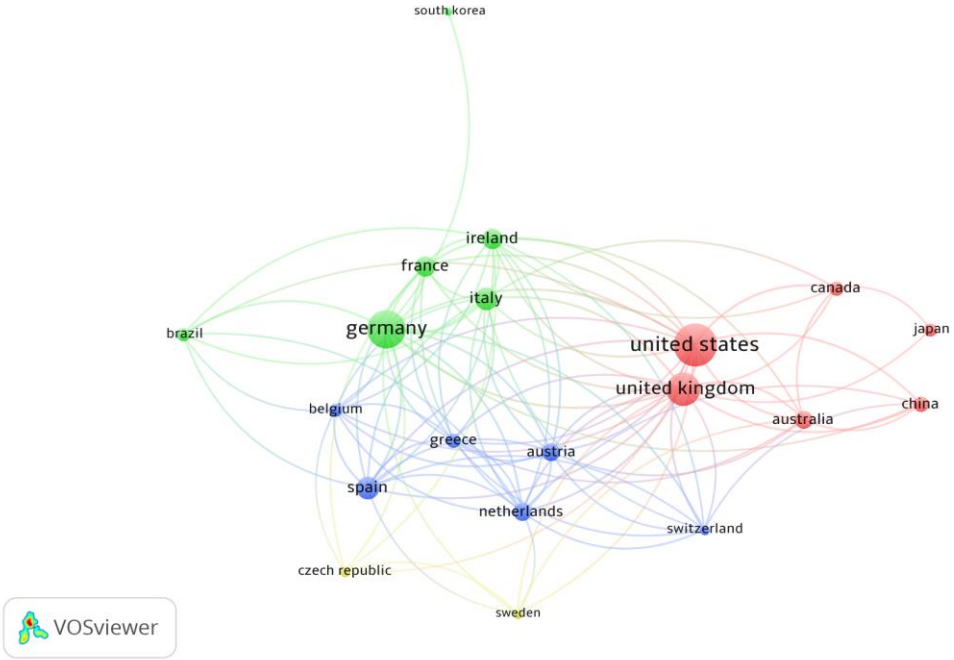

Ilustración 18. Mapa de coocurrencias de palabras clave de los autores de las publicaciones de Linked Data o Linked Open Data y bibliotecas en Scopus

B. ILUSTRACIONES Y TABLAS

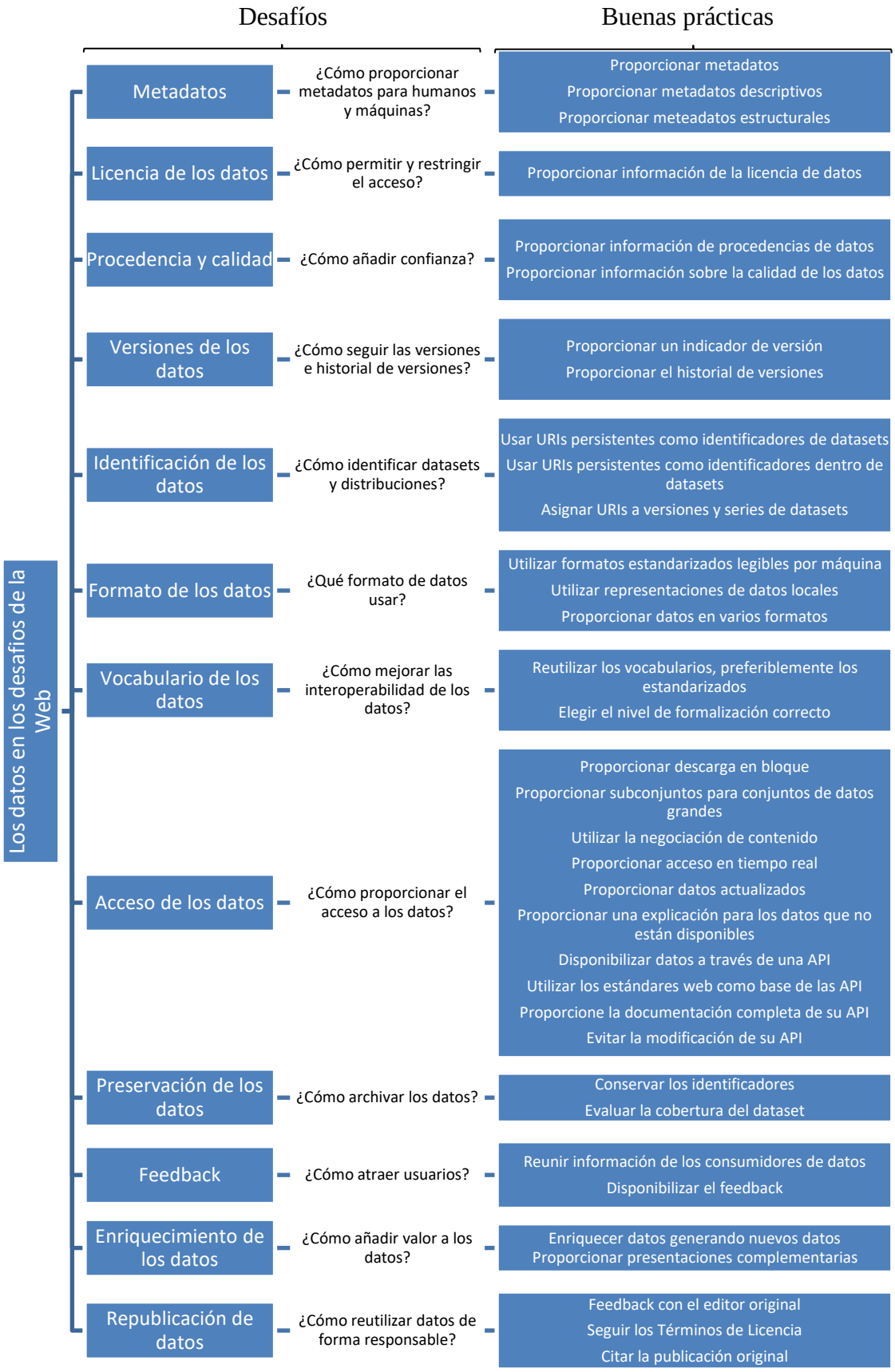

Ilustración 19. Desafíos y Buenas Prácticas del W3C (Lóscio, Burle y Calegari, 2016)

	1.Especificar	*2.Data Curation*	*3.Modelar*	*4.Generar*	*5.Enlazar*	*6.Publicar*	*7.Explorar*
Objetivos	Analizar y describir las características de los datos	Corregir y mejorar los datos fuente y RDF	Crear un vocabulario para describir los recursos RDF	Producir recursos RDF desde los datos fuente	Conectar los dataset RDF a otros datasets relevantes	Hacer disponible el dataset en la Web	Definir y desarrollar aplicaciones que hagan uso del dataset RDF
Subtareas	I. Identificar y analizar los datos fuente II. Asignar las URIs III. Definir las licencias e información de procedencia	I. Data sources curation II. RDF data curation	I. Analizar y seleccionar los vocabularios II. Desarrollar el vocabulario III. Vocabulario para representar la información de procedencia	I. Seleccionar extender o desarrollar las tecnologías para producir RDF II. Crear mapas entre los vocabularios y los datos fuente III. Transformar los datos fuente en RDF	I. Seleccionar los datasets objetivos para enlazar las entidades en los dataset II. Descubrir los enlaces con datasets objetivos III. Validar los enlaces	I. Publicar el dataset II. Publicar metadatos que describan los dataset III. Permitir el descubrimiento efectivo del dataset	I. Desarrollar o configurar aplicaciones sustentadas en el dataset

Tabla 8. Ciclo de vida para la publicación de Linked Data en la Biblioteca Nacional de España (Vila-Suero, Villazon-Terrazas y Gomez-Perez, 2013)

	Biblioteca Nacional de Francia	*Europeana*	*Library of Congress*	*British Library*	*Biblioteca Nacional de España*
FRBR	✓				✓
SKOS	✓	✓	✓	✓	
InterMarc	✓				
XMLEAD	✓				
Dublin Core	✓	✓	✓	✓	✓
RDF	✓	✓			✓
OAI_ORE		✓			
FOAF		✓	✓	✓	
BIBO			✓	✓	
BIO			✓	✓	
ISBD			✓	✓	✓
ORG			✓	✓	
RDF SCHEMA			✓	✓	
OWL			✓	✓	
WGS84			✓	✓	
RDA			✓		✓
FRAD					✓
FRSAD					✓
IFLA					✓
MADS/RDF					✓
DC					✓

Tabla 9. Vocabularios y ontologías usados por los grandes proyectos de Linked Data en bibliotecas

	Pantalla completa	Presentación abreviada	Endnote-Refworks	ProCite	MARC[31]
Signatura	✓	×	✓	✓	✓
Autor	✓	✓	✓	✓	✓
Titulo	✓	✓	✓	✓	✓
Edición	✓	×	✓	✓	✓
Lugar publicación	✓	×	✓	✓	✓
Fecha publicación	✓	×	✓	✓	✓
Publicación	✓	✓	✓	✓	✓
Nota	✓	×	✓	✓	✓
Materia	✓	×	✓	✓	✓
Autor secundario	✓	×	✓	✓	✓
ISBN	✓	×	✓	✓	✓
Lugar impresión	✓	×	✓	×	✓
Etiqueta	×	×	✓	×	✓
Resumen	✓	×	✓	✓	✓
Nombre de congreso	×	×	✓	✓	✓
Ubicación	✓	×	×	×	✓
Descripción	✓	×	×	×	✓
Colección	✓	×	×	×	✓
Titulo secundario	✓	×	×	×	✓
Titulo anterior	✓	×	×	×	✓
Titulo posterior	✓	×	×	×	✓
Depósito legal	✓	×	×	×	✓
CDU	✓	✓	×	×	✓
Bulletin	✓	×	×	×	✓
Términos	✓	×	×	×	✓
Nombre congreso	×	×	✓	✓	✓

Tabla 10. Comparativa de los campos que aparecen en los diferentes formatos de registros bibliográficos descargados desde Adrastea

[31] En color verde aparecen todos aquellos campos que figuran perfectamente, en naranja los que lo hacen pero dentro de un campo más general (por ejemplo, los datos relativos a fecha de publicación, editorial y lugar de publicación en Pantalla Completa lo hacen en un campo y Endnote-Refworks y ProCite en varios) y en rojo los que no lo hacen.

C. HERRAMIENTAS PARA LINKED DATA

Almacenamiento y gestión de datos

	Apache Hadoop
URL	http://hadoop.apache.org/
Descripción	Framework de software open-source para el almacenamiento distribuido de conjuntos de datos muy grandes en clusters de ordenadores. Requiere de conocimientos previos en el tratamiento de datos y en el uso de Java.
Licencia	Apache License 2.0
Plataformas	Multiplataforma

	Cloudera Distributed Hadoop (CDH)
URL	http://www.cloudera.com/downloads/cdh/5-9-0.html
Descripción	Distribución de Apache Hadoop, la cual conserva algunos elementos open-source y que está orientado al mundo empresarial.
Licencia	Apache License 2.0
Plataformas	Linux

	MongoDB
URL	https://www.mongodb.com/
Descripción	Se trata de la base de datos NoSQL más usada, bajo el modelo open-source, estando orientada a los documentos.
Licencia	GNU AGPL v3.0
Plataformas	Multiplataforma

Extracción y limpieza de datos

	Spoon - Pentaho's Data Integration
URL	http://community.pentaho.com/projects/data-integration/
Descripción	Herramienta open-source para la extracción, transformación, transporte y carga de datos (ETL).
Licencia	Apache License 2.0
Plataformas	Multiplataforma

	Virtuoso Sponger
URL	https://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/VirtSponger
Descripción	Se trata de un componente middleware de Virtuoso Open-Source (VOS) que permite importar datos en diversos formatos (CSV, RSS, vCard…) y transformarlos en RDF.
Licencia	GNU General Public License Version 2
Plataformas	Multiplataforma

	D2RQ
URL	http://d2rq.org/
Descripción	Sistema open-source que permite acceder a bases de datos relacionales como grafos RDF virtuales, pudiendo lanzar consultas SPARQL en bases de datos no RDF, así como exportar la base de datos en RDF.
Licencia	Apache License 2.0
Plataformas	Multiplataforma

	OpenRefine
URL	http://openrefine.org/
Descripción	Herramienta ETL (Extraer, Transformar y Cargar) enfocada a la limpieza, transformación, exploración y enlazado de datos procedentes de diversos formatos. Sus funciones se pueden expandir con el uso de extensiones, destacando RDF Refine extension o DBpedia extension.
Licencia	BSD
Plataformas	Multiplataforma

	GraphDB Free Edition
URL	http://ontotext.com/graphdb-8-enterprise-linked-data
Descripción	GraphDB es un repositorio semántico, un sistema de base de datos NoSQL que permite almacenar, consultar y gestionar datos estructurados. Utiliza ontologías para razonar automáticamente sobre los datos. Ofrece diferentes productos, entre los que se encuentra la versión gratuita, GraphDB Free, para colecciones de datos no demasiado extensas.
Licencia	Licencia libre de tipo RDBMS
Plataformas	Multiplataforma

Modelización

	Protégé
URL	http://protege.stanford.edu/
Descripción	Herramienta open-source que permite la construcción de modelos de dominio y aplicaciones basadas en el conocimiento con ontologías. Cuenta con una versión web y otra de escritorio. Es compatible con la última versión del Lenguaje de ontologías OWL 2 y especificaciones RDF de la World Wide Web Consortium (W3C).
Licencia	FreeBSD
Plataformas	Multiplataforma

	CmapTools Ontology Editor (COE)
URL	http://protege.stanford.edu/
Descripción	Versión de CmapTools, herramienta para los mapas conceptuales, orientada a construir, compartir y visualizar ontologías OWL.
Licencia	-
Plataformas	Multiplataforma

	OntoWiki
URL	http://ontowiki.net/
Descripción	Esta herramienta open-source permite la edición del contenido de archivos RDF de una forma muy visual, del mismo modo que un editor WYSIWIG para documentos de texto.
Licencia	GNU General Public License Version 2
Plataformas	Multiplataforma

	OOPS!
URL	http://oops.linkeddata.es/
Descripción	Se trata de una herramienta online de validación que permite detectar algunos de los errores más comunes que aparecen al desarrollar ontologías.
Licencia	-
Plataformas	Online

	W3C RDF Validation Service
URL	https://www.w3.org/RDF/Validator/
Descripción	Herramienta online de W3C para la validación y visualización de documentos RDF (RDF/XML).
Licencia	-
Plataformas	Online

Enlazado

	LIMES
URL	http://aksw.org/Projects/LIMES.html
Descripción	Framework que implementa métodos eficientes en tiempo para el descubrimiento de enlaces a gran escala basados en las características de los espacios métricos.
Licencia	GNU General Public License
Plataformas	Multiplataforma

	Silk
URL	http://silkframework.org/
Descripción	Open-source framework para combinar fuentes de datos heterogéneas, permitiendo generar enlaces entre elementos de datos contenidos en distintas fuentes.
Licencia	Apache License 2.0
Plataformas	Multiplataforma

Publicación

	Virtuoso Open-Source (VOS)
URL	https://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/
Descripción	Servidor multiplataforma escalable para el acceso a datos, integración y gestión de bases de datos relacionales, RDF y XML con un servidor de aplicaciones, de servicios Web.
Licencia	Apache License 2.0
Plataformas	Multiplataforma

D. APLICACIÓN JAVA

A través de NetBeans IDE 8.0.2 se ha realizado un pequeño programa para solucionar la falta de estructuración de los formatos de Pantalla completa y Endnote-Refworks proporcionados por la Biblioteca de la Universidad de Granada.

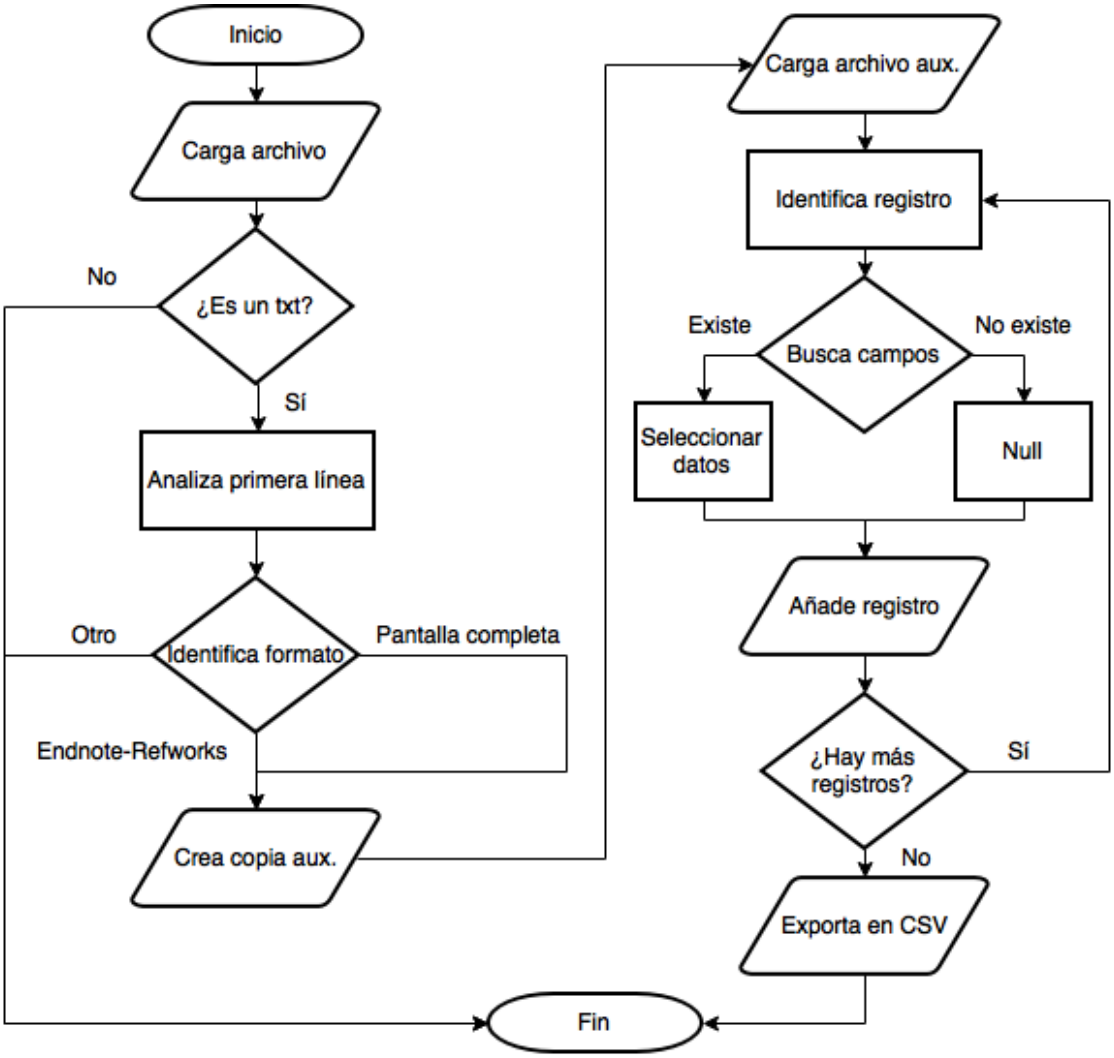

Ilustración 20 - Algoritmo de la aplicación realizada en Java

De manera general, el algoritmo de la aplicación (Ilustración 20) comienza importando un archivo, del cual analiza por medio de una sencilla expresión regular el nombre para saber si este es un archivo de texto (.*txt$). Una vez confirmado analiza la primera línea del contenido, existiendo tres posibilidades:

- La cadena de texto empieza por “Registro 1 de x” – Lo identifica como formato Pantalla completa y extrae el x como el número total de registros, transformando

la variable a tipo numérico. Recorre todo el texto, añade una línea extra al final del todo para evitar problemas después con su lectura por medio de bucles, crea una copia y lanza el método destinado a extraer los registros y campos de este, para lo cual envía el número de registros identificados.

- La cadena de texto empieza por "%" – lo identifica como Endnote-Refworks. Lo recorre de principio a final contando los campos título (%T), ya que todos los registros tendrán uno, y traduce ese número como el número total de registros existentes. Tras ello realiza los mismos pasos que en el anterior caso, pero con el método creado para Endnote-Refworks.
- No es ninguna de las opciones anteriores – No recorre más instrucciones y finaliza la aplicación.

Así mismo, para conocer cuál es el campo que representa cada abreviatura, en el caso de Endnote-Refworks, se encontró la equivalencia de cada etiqueta (Wikipedia, 2017), aunque existían varios campos que son completamente personalizables (%1, %2, %3, %4, %# y %$), por lo cual se localizó a que campo hacían referencia buscando en otros formatos.

Una vez se han recogido todos los datos de un registro, este es creado a través de otro método específico para cada uno de ello, añadiéndose la información reunida a un vector. Por último, gracias a una librería (MyUtilsJava, 2015), esos datos son leídos y exportados como un CSV.

A continuación se incluye el código fuente de esta aplicación:

```
001 package transformacsv;
002
003 import java.awt.Component;
004 import com.csvreader.CsvWriter;
005 import java.io.BufferedReader;
006 import java.io.BufferedWriter;
007 import java.io.File;
008 import java.io.FileInputStream;
009 import java.io.FileOutputStream;
010 import java.io.FileReader;
011 import java.io.FileWriter;
012 import java.io.IOException;
013 import java.io.InputStreamReader;
014 import java.io.OutputStreamWriter;
015 import java.util.ArrayList;
016 import java.util.List;
017 import java.util.regex.Matcher;
```

```
018 import java.util.regex.Pattern;
019 import javax.swing.JFileChooser;
020 import javax.swing.JOptionPane;
021
022 public class TransformaCSV {
023
024     public static void transformadorEndnote(int registros) {
025
026         File endnote = new File("copia.txt");
027
028         BufferedReader bufferendnote = null;
029
030         try {
031
032             bufferendnote = new BufferedReader(new
InputStreamReader(new FileInputStream(endnote), "ISO-8859-1"));
033
034             String autor = null, fecha = null, titulo = null,
autor2 = null, titulo2 = null, lpublicacion = null,
                    publicacion = null, lugarimpresi = null,
035 edicion = null, autorsub = null,
            isbn = null, nota = null, signatura = null, congreso =
036 null, file = null, materia = null, resumen = null;
037
038             List<CamposEndnote> tabla = new
ArrayList<CamposEndnote>();
039
040             String lineaendnote = null;
041             lineaendnote = bufferendnote.readLine();
042
043             System.out.println(lineaendnote);
044
045             for (int i = 1; i <= registros; i++) {
046
047                 if (lineaendnote.startsWith("%A")) {
048                     autor = lineaendnote.substring(3);
049                     lineaendnote = bufferendnote.readLine();
050                     while (lineaendnote.startsWith("%A")) {
                        autor = autor + "\t" +
051 lineaendnote.substring(3);
052                         lineaendnote = bufferendnote.readLine();
053                     }
054                 } else {
055                     autor = null;
056                 }
057
058                 if (lineaendnote.startsWith("%D")) {
059                     fecha = lineaendnote.substring(3);
060                     lineaendnote = bufferendnote.readLine();
061                     while (lineaendnote.startsWith("%D")) {
                        fecha = fecha + "\t" +
062 lineaendnote.substring(3);
063                         lineaendnote = bufferendnote.readLine();
064                     }
065                 } else {
066                     fecha = null;
067                 }
```

```
068
069                     if (lineaendnote.startsWith("%T")) {
070                         titulo = lineaendnote.substring(3);
071                         lineaendnote = bufferendnote.readLine();
072                         while (lineaendnote.startsWith("%T")) {
                                titulo = titulo + "\t" +
073 lineaendnote.substring(3);
074                             lineaendnote = bufferendnote.readLine();
075                         }
076                     } else {
077                         titulo = null;
078                     }
079
080                     if (lineaendnote.startsWith("%E")) {
081                         autor2 = lineaendnote.substring(3);
082                         lineaendnote = bufferendnote.readLine();
083                         while (lineaendnote.startsWith("%E")) {
                                autor2 = autor2 + "\t" +
084 lineaendnote.substring(3);
085                             lineaendnote = bufferendnote.readLine();
086                         }
087                     } else {
088                         autor2 = null;
089                     }
090
091                     if (lineaendnote.startsWith("%B")) {
092                         titulo2 = lineaendnote.substring(3);
093                         lineaendnote = bufferendnote.readLine();
094                         while (lineaendnote.startsWith("%B")) {
                                titulo2 = titulo2 + "\t" +
095 lineaendnote.substring(3);
096                             lineaendnote = bufferendnote.readLine();
097                         }
098                     } else {
099                         titulo2 = null;
100                     }
101
102                     if (lineaendnote.startsWith("%C")) {
103                         lpublicacion = lineaendnote.substring(3);
104                         lineaendnote = bufferendnote.readLine();
105                         while (lineaendnote.startsWith("%C")) {
                                lpublicacion = lpublicacion + "\t" +
106 lineaendnote.substring(3);
107                             lineaendnote = bufferendnote.readLine();
108                         }
109                     } else {
110                         lpublicacion = null;
111                     }
112
113                     if (lineaendnote.startsWith("%I")) {
114                         publicacion = lineaendnote.substring(3);
115                         lineaendnote = bufferendnote.readLine();
116                         while (lineaendnote.startsWith("%I")) {
                                publicacion = publicacion + "\t" +
117 lineaendnote.substring(3);
118                             lineaendnote = bufferendnote.readLine();
```

```
                        }
                    } else {
                        publicacion = null;
                    }

                    if (lineaendnote.startsWith("%V")) {
                        lugarimpresi = lineaendnote.substring(3);
                        lineaendnote = bufferendnote.readLine();
                        while (lineaendnote.startsWith("%V")) {
                            lugarimpresi = lugarimpresi + "\t" +
lineaendnote.substring(3);
                            lineaendnote = bufferendnote.readLine();
                        }
                    } else {
                        lugarimpresi = null;
                    }

                    if (lineaendnote.startsWith("%7")) {
                        edicion = lineaendnote.substring(3);
                        lineaendnote = bufferendnote.readLine();
                        while (lineaendnote.startsWith("%7")) {
                            edicion = edicion + "\t" +
lineaendnote.substring(3);
                            lineaendnote = bufferendnote.readLine();
                        }
                    } else {
                        edicion = null;
                    }

                    if (lineaendnote.startsWith("%?")) {
                        autorsub = lineaendnote.substring(3);
                        lineaendnote = bufferendnote.readLine();
                        while (lineaendnote.startsWith("%?")) {
                            autorsub = autorsub + "\t" +
lineaendnote.substring(3);
                            lineaendnote = bufferendnote.readLine();
                        }
                    } else {
                        autorsub = null;
                    }

                    if (lineaendnote.startsWith("%@")) {
                        isbn = lineaendnote.substring(3);
                        lineaendnote = bufferendnote.readLine();
                        while (lineaendnote.startsWith("%@")) {
                            isbn = isbn + "\t" +
lineaendnote.substring(3);
                            lineaendnote = bufferendnote.readLine();
                        }
                    } else {
                        isbn = null;
                    }

                    if (lineaendnote.startsWith("%2")) {
                        nota = lineaendnote.substring(3);
                        lineaendnote = bufferendnote.readLine();
```

```
                    while (lineaendnote.startsWith("%2")) {
                        nota = nota + "\t" +
lineaendnote.substring(3);
                        lineaendnote = bufferendnote.readLine();
                    }
                } else {
                    nota = null;
                }

                if (lineaendnote.startsWith("%3")) {
                    signatura = lineaendnote.substring(3);
                    lineaendnote = bufferendnote.readLine();
                    while (lineaendnote.startsWith("%3")) {
                        signatura = signatura + "\t" +
lineaendnote.substring(3);
                        lineaendnote = bufferendnote.readLine();
                    }
                }  else {
                    signatura = null;
                }

                if (lineaendnote.startsWith("%4")) {
                    congreso = lineaendnote.substring(3);
                    lineaendnote = bufferendnote.readLine();
                    while (lineaendnote.startsWith("%4")) {
                        congreso = congreso + "\t" +
lineaendnote.substring(3);
                        lineaendnote = bufferendnote.readLine();
                    }
                } else {
                    congreso = null;
                }

                if (lineaendnote.startsWith("%F")) {
                    file = lineaendnote.substring(3);
                    lineaendnote = bufferendnote.readLine();
                    while (lineaendnote.startsWith("%F")) {
                        file = file + "\t" +
lineaendnote.substring(3);
                        lineaendnote = bufferendnote.readLine();
                    }
                } else {
                    file = null;
                }

                if (lineaendnote.startsWith("%K")) {
                    materia = lineaendnote.substring(3);
                    lineaendnote = bufferendnote.readLine();
                    while (lineaendnote.startsWith("%K")) {
                        materia = materia + "\t" +
lineaendnote.substring(3);
                        lineaendnote = bufferendnote.readLine();
                    }
                } else {
                    materia = null;
                }
```

```
222
223                 if (lineaendnote.startsWith("%X")) {
224                     resumen = lineaendnote.substring(3);
225                     lineaendnote = bufferendnote.readLine();
226                     while (lineaendnote.startsWith("%X")) {
227                         resumen = resumen + "\t" +
lineaendnote.substring(3);
228                         lineaendnote = bufferendnote.readLine();
229                     }
230                 } else {
231                     resumen = null;
232                 }
233
234                 lineaendnote = bufferendnote.readLine();
235
236                 tabla.add(new CamposEndnote(autor, fecha, titulo,
autor2, titulo2, lpublicacion,
237                         publicacion, lugarimpresi, edicion,
autorsub, isbn, nota, signatura, congreso, file,
238                         materia, resumen));
239
240             }
241
242             //Para evitar más ficheros de los necesarios se
eliminará el auxiliar
243             /*if (endnote.delete()) {
244              System.out.println("El fichero auxiliar fue
borrado");
245              } else {
246              System.out.println("El fichero auxiliar no se pudo
borrar");
247              }*/
248             //Se va a verificar que el archivo csv no existe
249             String nombre = "endnote.csv";
250             boolean existe = new File(nombre).exists();
251
252             if (existe) {
253                 File archivoendnote = new File(nombre);
254                 archivoendnote.delete();
255             }
256
257             try {
258
259                 CsvWriter csvendnote = new CsvWriter(new
FileWriter(nombre, true), ',');
260
261                 csvendnote.write("Autor");
262                 csvendnote.write("Fecha");
263                 csvendnote.write("Titulo");
264                 csvendnote.write("Autor secundario");
265                 csvendnote.write("Título secundario");
266                 csvendnote.write("L. publicacion");
267                 csvendnote.write("Publicacion");
268                 csvendnote.write("Lugar impresion");
269                 csvendnote.write("Edicion");
270                 csvendnote.write("Autor subsidiario");
271                 csvendnote.write("ISBN");
```

```
                    csvendnote.write("Nota");
                    csvendnote.write("Signatura");
                    csvendnote.write("Congreso");
                    csvendnote.write("Etiqueta");
                    csvendnote.write("Materia");
                    csvendnote.write("Resumen");

                    csvendnote.endRecord();

                    for (CamposEndnote tab : tabla) {

                        csvendnote.write(tab.getAutor());
                        csvendnote.write(tab.getFecha());
                        csvendnote.write(tab.getTitulo());
                        csvendnote.write(tab.getAutor2());
                        csvendnote.write(tab.getTitulo2());
                        csvendnote.write(tab.getLpublicacion());
                        csvendnote.write(tab.getPublicacion());
                        csvendnote.write(tab.getLugarimpresi());
                        csvendnote.write(tab.getEdicion());
                        csvendnote.write(tab.getAutorsub());
                        csvendnote.write(tab.getIsbn());
                        csvendnote.write(tab.getNota());
                        csvendnote.write(tab.getSignatura());
                        csvendnote.write(tab.getCongreso());
                        csvendnote.write(tab.getFile());
                        csvendnote.write(tab.getMateria());
                        csvendnote.write(tab.getResumen());
                        csvendnote.endRecord();
                    }

                    csvendnote.close();

                } catch (IOException e) {
                    e.printStackTrace();
                }

            } catch (Exception e) {
                e.printStackTrace();
            } finally {
                try {
                    if (null != bufferendnote) {
                        System.out.println("cerramos");
                        bufferendnote.close();
                    }
                } catch (Exception e2) {
                    e2.printStackTrace();
                }

            }

        }

        public static void transformadorPantallacompleta(int
registros) {

```

```
327         File pantalla = new File("copia.txt");
328         BufferedReader bufferpantalla = null;
329 
330         try {
331 
332             bufferpantalla = new BufferedReader(new
InputStreamReader(new FileInputStream(pantalla), "ISO-8859-1"));
333 
334             String registro = null, ubicacion = null, autor =
null, titulo = null,
335                     edicion = null, publicacion = null,
descripcion = null, coleccion = null,
336                     nota = null, materia = null, autor2 = null,
titulo2 = null,
337                     titanter = null, titposter = null, isbn =
null, deposito = null,
338                     cdu = null, lugarimpresi = null, bulletin =
null, terminos = null,
339                     contador = null;
340 
341             int j = 0, k = 1;
342 
343             List<CamposPantalla> tabla = new
ArrayList<CamposPantalla>();
344 
345             String lineapantalla = null;
346 
347 
348 
349             for (int i = 1; i <= registros; i++) {
350 
351                 lineapantalla = bufferpantalla.readLine();
352                 if (lineapantalla.startsWith("Registro " + k + "
de 1251")) {
353                     registro = Integer.toString(k);
354                     k++;
355                     lineapantalla = bufferpantalla.readLine();
356                 }
357 
358                 if (lineapantalla.startsWith("UBICACION")) {
359                     ubicacion = lineapantalla.substring(13);
360                     lineapantalla = bufferpantalla.readLine();
361                     while (lineapantalla.startsWith("
")) {
362                         ubicacion = ubicacion +
lineapantalla.substring(15);
363                         lineapantalla = bufferpantalla.readLine();
364                     }
365                     while (lineapantalla.startsWith("UBICACION"))
{
366                         ubicacion = ubicacion + "\t" +
lineapantalla.substring(13);
367                         lineapantalla = bufferpantalla.readLine();
368                         while (lineapantalla.startsWith("
")) {
369                             ubicacion = ubicacion +
lineapantalla.substring(15);
370                             lineapantalla =
```

```
bufferpantalla.readLine();
                            }
                        }
                    } else {
                        ubicacion = null;
                    }

                    if (lineapantalla.startsWith("AUTOR")) {
                        autor = lineapantalla.substring(13);
                        lineapantalla = bufferpantalla.readLine();
                        while (lineapantalla.startsWith("
")) {
                            autor = autor +
lineapantalla.substring(15);
                            lineapantalla = bufferpantalla.readLine();
                        }
                        while (lineapantalla.startsWith("AUTOR")) {
                            autor = autor + "\t" +
lineapantalla.substring(13);
                            lineapantalla = bufferpantalla.readLine();
                            while (lineapantalla.startsWith("
")) {
                                autor = autor +
lineapantalla.substring(15);
                                lineapantalla =
bufferpantalla.readLine();
                            }
                        }
                    } else {
                        autor = null;
                    }

                    if (lineapantalla.startsWith("TÍTULO")) {
                        titulo = lineapantalla.substring(13);
                        lineapantalla = bufferpantalla.readLine();
                        while (lineapantalla.startsWith("
")) {
                            titulo = titulo +
lineapantalla.substring(15);
                            lineapantalla = bufferpantalla.readLine();
                        }
                        while (lineapantalla.startsWith("TÍTULO")) {
                            titulo = titulo + "\t" +
lineapantalla.substring(13);
                            lineapantalla = bufferpantalla.readLine();
                            while (lineapantalla.startsWith("
")) {
                                titulo = titulo +
lineapantalla.substring(15);
                                lineapantalla =
bufferpantalla.readLine();
                            }
                        }
                    } else {
                        titulo = null;
                    }

```

```
                if (lineapantalla.startsWith("EDICIÓN")) {
                    edicion = lineapantalla.substring(13);
                    lineapantalla = bufferpantalla.readLine();
                    while (lineapantalla.startsWith("
")) {
                        edicion = edicion +
lineapantalla.substring(15);
                        lineapantalla = bufferpantalla.readLine();
                    }
                    while (lineapantalla.startsWith("EDICIÓN")) {
                        edicion = edicion + "\t" +
lineapantalla.substring(13);
                        lineapantalla = bufferpantalla.readLine();
                        while (lineapantalla.startsWith("
")) {
                            edicion = edicion +
lineapantalla.substring(15);
                            lineapantalla =
bufferpantalla.readLine();
                        }
                    }
                } else {
                    edicion = null;
                }

                if (lineapantalla.startsWith("PUBLICAC")) {
                    publicacion = lineapantalla.substring(13);
                    lineapantalla = bufferpantalla.readLine();
                    while (lineapantalla.startsWith("
")) {
                        publicacion = publicacion +
lineapantalla.substring(15);
                        lineapantalla = bufferpantalla.readLine();
                    }
                    while (lineapantalla.startsWith("PUBLICAC")) {
                        publicacion = publicacion + "\t" +
lineapantalla.substring(13);
                        lineapantalla = bufferpantalla.readLine();
                        while (lineapantalla.startsWith("
")) {
                            publicacion = publicacion +
lineapantalla.substring(15);
                            lineapantalla =
bufferpantalla.readLine();
                        }
                    }
                } else {
                    publicacion = null;
                }

                if (lineapantalla.startsWith("DESCRIPCIÓN")) {
                    descripcion = lineapantalla.substring(13);
                    lineapantalla = bufferpantalla.readLine();
                    while (lineapantalla.startsWith("
")) {
                        descripcion = descripcion +
lineapantalla.substring(15);
                        lineapantalla = bufferpantalla.readLine();
```

```
459                     }
460                     while
(lineapantalla.startsWith("DESCRIPCIÓN")) {
461                         descripcion = descripcion + "\t" +
lineapantalla.substring(13);
462                         lineapantalla = bufferpantalla.readLine();
463                         while (lineapantalla.startsWith("
")) {
464                             descripcion = descripcion +
lineapantalla.substring(15);
465                             lineapantalla =
bufferpantalla.readLine();
466                         }
467                     }
468                 } else {
469                     descripcion = null;
470                 }
471
472                 if (lineapantalla.startsWith("COLECCIÓN")) {
473                     coleccion = lineapantalla.substring(13);
474                     lineapantalla = bufferpantalla.readLine();
475                     while (lineapantalla.startsWith("
")) {
476                         coleccion = coleccion +
lineapantalla.substring(15);
477                         lineapantalla = bufferpantalla.readLine();
478                     }
479                     while (lineapantalla.startsWith("COLECCIÓN"))
{
480                         coleccion = coleccion + "\t" +
lineapantalla.substring(13);
481                         lineapantalla = bufferpantalla.readLine();
482                         while (lineapantalla.startsWith("
")) {
483                             coleccion = coleccion +
lineapantalla.substring(15);
484                             lineapantalla =
bufferpantalla.readLine();
485                         }
486                     }
487                 } else {
488                     coleccion = null;
489                 }
490
491                 if (lineapantalla.startsWith("NOTE")) {
492                     nota = lineapantalla.substring(13);
493                     lineapantalla = bufferpantalla.readLine();
494                     while (lineapantalla.startsWith("
")) {
495                         nota = nota + lineapantalla.substring(15);
496                         lineapantalla = bufferpantalla.readLine();
497                     }
498                     while (lineapantalla.startsWith("NOTE")) {
499                         nota = nota + "\t" +
lineapantalla.substring(13);
500                         lineapantalla = bufferpantalla.readLine();
501                         while (lineapantalla.startsWith("
")) {
```

```
                                nota = nota +
lineapantalla.substring(15);
                                lineapantalla =
bufferpantalla.readLine();
                            }
                        }
                    } else {
                        nota = null;
                    }

                    if (lineapantalla.startsWith("MATERIA")) {
                        materia = lineapantalla.substring(13);
                        lineapantalla = bufferpantalla.readLine();
                        while (lineapantalla.startsWith("
")) {
                            materia = materia +
lineapantalla.substring(15);
                            lineapantalla = bufferpantalla.readLine();
                        }
                        while (lineapantalla.startsWith("MATERIA")) {
                            materia = materia + "\t" +
lineapantalla.substring(13);
                            lineapantalla = bufferpantalla.readLine();
                            while (lineapantalla.startsWith("
")) {
                                materia = materia +
lineapantalla.substring(15);
                                lineapantalla =
bufferpantalla.readLine();
                            }
                        }
                    } else {
                        materia = null;
                    }

                    if (lineapantalla.startsWith("AUTOR SEC")) {
                        autor2 = lineapantalla.substring(13);
                        lineapantalla = bufferpantalla.readLine();
                        while (lineapantalla.startsWith("
")) {
                            autor2 = autor2 +
lineapantalla.substring(15);
                            lineapantalla = bufferpantalla.readLine();
                        }
                        while (lineapantalla.startsWith("AUTOR SEC"))
{
                            autor2 = autor2 + "\t" +
lineapantalla.substring(13);
                            lineapantalla = bufferpantalla.readLine();
                            while (lineapantalla.startsWith("
")) {
                                autor2 = autor2 +
lineapantalla.substring(15);
                                lineapantalla =
bufferpantalla.readLine();
                            }
                        }
                    } else {
```

```
                        autor2 = null;
                    }

                    if (lineapantalla.startsWith("TÍTULO SEC")) {
                        titulo2 = lineapantalla.substring(13);
                        lineapantalla = bufferpantalla.readLine();
                        while (lineapantalla.startsWith("
")) {
                            titulo2 = titulo2 +
lineapantalla.substring(15);
                            lineapantalla = bufferpantalla.readLine();
                        }
                        while (lineapantalla.startsWith("TÍTULO SEC"))
{
                            titulo2 = titulo2 + "\t" +
lineapantalla.substring(13);
                            lineapantalla = bufferpantalla.readLine();
                            while (lineapantalla.startsWith("
")) {
                                titulo2 = titulo2 +
lineapantalla.substring(15);
                                lineapantalla =
bufferpantalla.readLine();
                            }
                        }
                    } else {
                        titulo2 = null;
                    }

                    if (lineapantalla.startsWith("TIT ANTER")) {
                        titanter = lineapantalla.substring(13);
                        lineapantalla = bufferpantalla.readLine();
                        while (lineapantalla.startsWith("
")) {
                            titanter = titanter +
lineapantalla.substring(15);
                            lineapantalla = bufferpantalla.readLine();
                        }
                        while (lineapantalla.startsWith("TIT ANTER"))
{
                            titanter = titanter + "\t" +
lineapantalla.substring(13);
                            lineapantalla = bufferpantalla.readLine();
                            while (lineapantalla.startsWith("
")) {
                                titanter = titanter +
lineapantalla.substring(15);
                                lineapantalla =
bufferpantalla.readLine();
                            }
                        }
                    } else {
                        titanter = null;
                    }

                    if (lineapantalla.startsWith("TIT POSTER")) {
                        titposter = lineapantalla.substring(13);
                        lineapantalla = bufferpantalla.readLine();
```

```
589                     while (lineapantalla.startsWith("
"))  {
590                         titposter = titposter +
lineapantalla.substring(15);
591                         lineapantalla = bufferpantalla.readLine();
592                     }
593                     while (lineapantalla.startsWith("TIT POSTER"))
{
594                         titposter = titposter + "\t" +
lineapantalla.substring(13);
595                         lineapantalla = bufferpantalla.readLine();
596                         while (lineapantalla.startsWith("
"))  {
597                             titposter = titposter +
lineapantalla.substring(15);
598                             lineapantalla =
bufferpantalla.readLine();
599                         }
600                     }
601                 } else {
602                     titposter = null;
603                 }
604 
605                 if (lineapantalla.startsWith("ISBN"))  {
606                     isbn = lineapantalla.substring(13);
607                     lineapantalla = bufferpantalla.readLine();
608                     while (lineapantalla.startsWith("
"))  {
609                         isbn = isbn + lineapantalla.substring(15);
610                         lineapantalla = bufferpantalla.readLine();
611                     }
612                     while (lineapantalla.startsWith("ISBN"))  {
613                         isbn = isbn + "\t" +
lineapantalla.substring(13);
614                         lineapantalla = bufferpantalla.readLine();
615                         while (lineapantalla.startsWith("
"))  {
616                             isbn = isbn +
lineapantalla.substring(15);
617                             lineapantalla =
bufferpantalla.readLine();
618                         }
619                     }
620                 } else {
621                     isbn = null;
622                 }
623 
624                 if (lineapantalla.startsWith("DEP. LEGAL"))  {
625                     deposito = lineapantalla.substring(13);
626                     lineapantalla = bufferpantalla.readLine();
627                     while (lineapantalla.startsWith("
"))  {
628                         deposito = deposito +
lineapantalla.substring(15);
629                         lineapantalla = bufferpantalla.readLine();
630                     }
631                     while (lineapantalla.startsWith("DEP. LEGAL"))
{
```

```
632                             deposito = deposito + "\t" +
lineapantalla.substring(13);
633                             lineapantalla = bufferpantalla.readLine();
634                             while (lineapantalla.startsWith("
")) {
635                                 deposito = deposito +
lineapantalla.substring(15);
636                                 lineapantalla =
bufferpantalla.readLine();
637                             }
638                         }
639                     } else {
640                         deposito = null;
641                     }
642 
643                     if (lineapantalla.startsWith("CDU")) {
644                         cdu = lineapantalla.substring(13);
645                         lineapantalla = bufferpantalla.readLine();
646                         while (lineapantalla.startsWith("
")) {
647                             cdu = cdu + lineapantalla.substring(15);
648                             lineapantalla = bufferpantalla.readLine();
649                         }
650                         while (lineapantalla.startsWith("CDU")) {
651                             cdu = cdu + "\t" +
lineapantalla.substring(13);
652                             lineapantalla = bufferpantalla.readLine();
653                             while (lineapantalla.startsWith("
")) {
654                                 cdu = cdu +
lineapantalla.substring(15);
655                                 lineapantalla =
bufferpantalla.readLine();
656                             }
657                         }
658                     } else {
659                         cdu = null;
660                     }
661 
662                     if (lineapantalla.startsWith("LUGAR IMPRESI")) {
663                         lugarimpresi = lineapantalla.substring(13);
664                         lineapantalla = bufferpantalla.readLine();
665                         while (lineapantalla.startsWith("
")) {
666                             lugarimpresi = lugarimpresi +
lineapantalla.substring(15);
667                             lineapantalla = bufferpantalla.readLine();
668                         }
669                         while (lineapantalla.startsWith("LUGAR
IMPRESI")) {
670                             lugarimpresi = lugarimpresi + "\t" +
lineapantalla.substring(13);
671                             lineapantalla = bufferpantalla.readLine();
672                             while (lineapantalla.startsWith("
")) {
673                                 lugarimpresi = lugarimpresi +
lineapantalla.substring(15);
674                                 lineapantalla =
```

```
bufferpantalla.readLine();
                        }
                    }
                } else {
                    lugarimpresi = null;
                }

                if (lineapantalla.startsWith("BULLETIN")) {
                    bulletin = lineapantalla.substring(13);
                    lineapantalla = bufferpantalla.readLine();
                    while (lineapantalla.startsWith("
")) {
                        bulletin = bulletin +
lineapantalla.substring(15);
                        lineapantalla = bufferpantalla.readLine();
                    }
                    while (lineapantalla.startsWith("BULLETIN")) {
                        bulletin = bulletin + "\t" +
lineapantalla.substring(13);
                        lineapantalla = bufferpantalla.readLine();
                        while (lineapantalla.startsWith("
")) {
                            bulletin = bulletin +
lineapantalla.substring(15);
                            lineapantalla =
bufferpantalla.readLine();
                        }
                    }
                } else {
                    bulletin = null;
                }

                if (lineapantalla.startsWith("TERMINOS")) {
                    terminos = lineapantalla.substring(13);
                    lineapantalla = bufferpantalla.readLine();
                } else {
                    terminos = null;
                }

                tabla.add(new CamposPantalla(registro, ubicacion,
autor, titulo, edicion, publicacion,
                        descripcion, coleccion, nota, materia,
autor2, titulo2, titanter,
                        titposter, isbn, deposito, cdu,
lugarimpresi, bulletin, terminos));

                //Para ir sorteando las líneas en las que parecen
las copias
                j = 1;
                contador = Integer.toString(j);
                while (lineapantalla.startsWith(contador) ||
lineapantalla.startsWith("0" + contador)) {
                    j = j + 1;
                    contador = Integer.toString(j);
                    lineapantalla = bufferpantalla.readLine();
                }

```

```
720             }
721
722             bufferpantalla.close();
723
724             //Para evitar más ficheros de los necesarios se
eliminará el auxiliar
725             if (pantalla.delete()) {
726                 System.out.println("El fichero auxiliar fue
borrado");
727             } else {
728                 System.out.println("El fichero auxiliar no se pudo
borrar");
729             }
730
731             //Se va a verificar que el archivo csv no existe
732             String nombre = "pantalla.csv";
733             boolean existe = new File(nombre).exists();
734
735             if (existe) {
736                 File archivopantalla = new File(nombre);
737                 archivopantalla.delete();
738             }
739
740             try {
741
742                 CsvWriter csvpantalla = new CsvWriter(new
FileWriter(nombre, true), ',');
743
744                 csvpantalla.write("Registro");
745                 csvpantalla.write("Ubicacion");
746                 csvpantalla.write("Autor");
747                 csvpantalla.write("Titulo");
748                 csvpantalla.write("Edicion");
749                 csvpantalla.write("Publicacion");
750                 csvpantalla.write("Descripcion");
751                 csvpantalla.write("Coleccion");
752                 csvpantalla.write("Nota");
753                 csvpantalla.write("Materia");
754                 csvpantalla.write("Autor secundario");
755                 csvpantalla.write("Titulo secundario");
756                 csvpantalla.write("Titulo anterior");
757                 csvpantalla.write("Titulo posterior");
758                 csvpantalla.write("ISBN");
759                 csvpantalla.write("Deposito legal");
760                 csvpantalla.write("CDU");
761                 csvpantalla.write("Lugar impresion");
762                 csvpantalla.write("Bulletin");
763                 csvpantalla.write("Terminos");
764
765                 csvpantalla.endRecord();
766
767                 for (CamposPantalla tab : tabla) {
768
769                     csvpantalla.write(tab.getRegistro());
770                     csvpantalla.write(tab.getUbicacion());
771                     csvpantalla.write(tab.getAutor());
```

```
                    csvpantalla.write(tab.getTitulo());
                    csvpantalla.write(tab.getEdicion());
                    csvpantalla.write(tab.getPublicacion());
                    csvpantalla.write(tab.getDescripcion());
                    csvpantalla.write(tab.getColeccion());
                    csvpantalla.write(tab.getNota());
                    csvpantalla.write(tab.getMateria());
                    csvpantalla.write(tab.getAutor2());
                    csvpantalla.write(tab.getTitulo2());
                    csvpantalla.write(tab.getTitanter());
                    csvpantalla.write(tab.getTitposter());
                    csvpantalla.write(tab.getIsbn());
                    csvpantalla.write(tab.getDeposito());
                    csvpantalla.write(tab.getCdu());
                    csvpantalla.write(tab.getLugarimpresi());
                    csvpantalla.write(tab.getBulletin());
                    csvpantalla.write(tab.getTerminos());
                    csvpantalla.endRecord();

                }

                csvpantalla.close();

            } catch (IOException e) {
                e.printStackTrace();
            }

        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (null != bufferpantalla) {
                    System.out.println("cerramos");
                    bufferpantalla.close();
                }
            } catch (Exception e2) {
                e2.printStackTrace();
            }

        }

    }

    public static void main(String[] args) {

        /*
         Preparación del fichero
         Se va a cargar el archivo a través de una ventana de
navegación
         */
        JFileChooser selector = new JFileChooser();
        Component ventana = null;
        File archivo = null;
        BufferedReader buffer = null;

        int seleccion = selector.showDialog(ventana, "Abrir");
```

```
827
828         if (seleccion == JFileChooser.APPROVE_OPTION) {
829
830             archivo = selector.getSelectedFile();
831
832             //Comprobamos que el archivo seleccionado sea un txt
833             Pattern expresion = Pattern.compile(".*txt$|.*TXT$");
834             Matcher compara =
expresion.matcher(selector.getName(archivo));
835
836             if (compara.matches()) {
837                 try {
838                     buffer = new BufferedReader(new
InputStreamReader(new FileInputStream(archivo), "UTF-8"));
839                     //Se va a leer el archivo y copiarlo a uno
nuevo
840                     String original = null, copia = "";
841                     int registros = 0;
842                     String tipo = null;
843
844
845                     //Se lee la primera línea para diferenciar
entre los dos formatos
846                     //Al crear el jar se crean tres caracteres y
hay que evitarlos
847                     original = buffer.readLine().substring(1);
848
849                     if (original.startsWith("%")) {
850
851                         tipo = "endnote";
852                         JOptionPane.showMessageDialog(ventana, "El
archivo seleccionado"
853                                 + " es un End-Note", "End-Note",
JOptionPane.INFORMATION_MESSAGE);
854
855                         do {
856                             //Para contar los registros se
contarán los campos título
857                             if (original.startsWith("%T")) {
858                                 registros++;
859                             }
860                             copia = copia + original + "\r\n";
861                         } while ((original = buffer.readLine()) !=
null);
862
863                     } else if (original.startsWith("Registro 1 de
")) {
864
865                         tipo = "pantalla";
866                         JOptionPane.showMessageDialog(ventana, "El
archivo seleccionado"
867                                 + " es Pantalla completa",
"Pantalla completa", JOptionPane.INFORMATION_MESSAGE);
868
869                         /*
870                          Para contar los registros se extrae la
cadena que lo índica
```

```
                            y se cambia a tipo int
                            */
                           registros =
Integer.parseInt(original.substring(14));
                           do {
                               copia = copia + original + "\r\n";
                           } while ((original = buffer.readLine()) !=
null);

                       }

                       JOptionPane.showMessageDialog(ventana, "Se han
encontrado "
                               + registros + " registros",
"Registros", JOptionPane.INFORMATION_MESSAGE);

                       //Se añade una línea en blanco de más para
evitar errores después
                       copia = copia + " \r\n";

                       try {

                           FileOutputStream archivocopia = new
FileOutputStream("copia.txt");
                           archivocopia.write(copia.getBytes());
                           archivocopia.flush();
                           archivocopia.close();
                       } catch (Exception e) {
                           System.out.println(e.getMessage());
                       }

                       if (tipo == "endnote") {
                           transformadorEndnote(registros);
                       } else {
                           transformadorPantallacompleta(registros);
                       }

                       JOptionPane.showMessageDialog(ventana,
"¡Transformación realizada con éxito!", "Registros",
JOptionPane.INFORMATION_MESSAGE);


                   } catch (Exception e) {
                       e.printStackTrace();
                   } finally {
                       try {
                           if (null != buffer) {
                               System.out.println("cerramos");
                               buffer.close();
                           }
                       } catch (Exception e2) {
                           e2.printStackTrace();
                       }

                   }
               } else {
                   JOptionPane.showMessageDialog(null, "El archivo
```

```
seleccionado no "
                            + "es correcto", "Error",
JOptionPane.WARNING_MESSAGE);
            }

        }

    }
}
```

Tabla 11. Clase principal de la aplicación Java

```
package transformacsv;

public class CamposPantalla {

    private String registro;
    private String ubicacion;
    private String autor;
    private String titulo;
    private String edicion;
    private String publicacion;
    private String descripcion;
    private String coleccion;
    private String nota;
    private String materia;
    private String autor2;
    private String titulo2;
    private String titanter;
    private String titposter;
    private String isbn;
    private String deposito;
    private String cdu;
    private String lugarimpresi;
    private String bulletin;
    private String terminos;

    public CamposPantalla(String registro, String ubicacion,
String autor, String titulo,
            String edicion, String publicacion, String
descripcion, String coleccion,
            String nota, String materia, String autor2, String
titulo2, String titanter,
            String titposter, String isbn, String deposito, String
cdu, String lugarimpresi,
            String bulletin, String terminos) {
        this.registro = registro;
        this.ubicacion = ubicacion;
        this.autor = autor;
        this.titulo = titulo;
        this.edicion = edicion;
        this.publicacion = publicacion;
        this.descripcion = descripcion;
        this.coleccion = coleccion;
        this.nota = nota;
        this.materia = materia;
```

```
041             this.autor2 = autor2;
042             this.titulo2 = titulo2;
043             this.titanter = titanter;
044             this.titposter = titposter;
045             this.isbn = isbn;
046             this.deposito = deposito;
047             this.cdu = cdu;
048             this.lugarimpresi = lugarimpresi;
049             this.bulletin = bulletin;
050             this.terminos = terminos;
051         }
052 
053         public String getRegistro() {
054             return registro;
055         }
056 
057         public void setRegistro(String registro) {
058             this.registro = registro;
059         }
060 
061         public String getUbicacion() {
062             return ubicacion;
063         }
064 
065         public void setUbicacion(String ubicacion) {
066             this.ubicacion = ubicacion;
067         }
068 
069         public String getAutor() {
070             return autor;
071         }
072 
073         public void setAutor(String autor) {
074             this.autor = autor;
075         }
076 
077         public String getTitulo() {
078             return titulo;
079         }
080 
081         public void setTitulo(String titulo) {
082             this.titulo = titulo;
083         }
084 
085         public String getEdicion() {
086             return edicion;
087         }
088 
089         public void setEdicion(String edicion) {
090             this.edicion = edicion;
091         }
092 
093         public String getPublicacion() {
094             return publicacion;
095         }
096 
```

```
097     public void setPublicacion(String publicacion) {
098         this.publicacion = publicacion;
099     }
100 
101     public String getDescripcion() {
102         return descripcion;
103     }
104 
105     public void setDescripcion(String descripcion) {
106         this.descripcion = descripcion;
107     }
108 
109     public String getColeccion() {
110         return coleccion;
111     }
112 
113     public void setColeccion(String coleccion) {
114         this.coleccion = coleccion;
115     }
116 
117     public String getNota() {
118         return nota;
119     }
120 
121     public void setNota(String nota) {
122         this.nota = nota;
123     }
124 
125     public String getMateria() {
126         return materia;
127     }
128 
129     public void setMateria(String materia) {
130         this.materia = materia;
131     }
132 
133     public String getAutor2() {
134         return autor2;
135     }
136 
137     public void setAutor2(String autor2) {
138         this.autor2 = autor2;
139     }
140 
141     public String getTitulo2() {
142         return titulo2;
143     }
144 
145     public void setTitulo2(String titulo2) {
146         this.titulo2 = titulo2;
147     }
148 
149     public String getTitanter() {
150         return titanter;
151     }
152 
```

```
153     public void setTitanter(String titanter) {
154         this.titanter = titanter;
155     }
156 
157     public String getTitposter() {
158         return titposter;
159     }
160 
161     public void setTitposter(String titposter) {
162         this.titposter = titposter;
163     }
164 
165     public String getIsbn() {
166         return isbn;
167     }
168 
169     public void setIsbn(String isbn) {
170         this.isbn = isbn;
171     }
172 
173     public String getDeposito() {
174         return deposito;
175     }
176 
177     public void setDeposito(String deposito) {
178         this.deposito = deposito;
179     }
180 
181     public String getCdu() {
182         return cdu;
183     }
184 
185     public void setCdu(String cdu) {
186         this.cdu = cdu;
187     }
188 
189     public String getLugarimpresi() {
190         return lugarimpresi;
191     }
192 
193     public void setLugarimpresi(String lugarimpresi) {
194         this.lugarimpresi = lugarimpresi;
195     }
196 
197     public String getBulletin() {
198         return bulletin;
199     }
200 
201     public void setBulletin(String bulletin) {
202         this.bulletin = bulletin;
203     }
204 
205     public String getTerminos() {
206         return terminos;
207     }
208 
```

```
209     public void setTerminos(String terminos) {
210         this.terminos = terminos;
211     }
212 
213 }
```

Tabla 12. Clase con los campos de Pantalla Completa en Java

```
001 package transformacsv;
002 
003 public class CamposEndnote {
004 
005     private String fecha;
006     private String autor;
007     private String titulo;
008     private String autor2;
009     private String titulo2;
010     private String lpublicacion;
011     private String publicacion;
012     private String lugarimpresi;
013     private String edicion;
014     private String autorsub;
015     private String isbn;
016     private String nota;
017     private String signatura;
018     private String congreso;
019     private String file;
020     private String materia;
021     private String resumen;
022 
023     public CamposEndnote(String autor, String fecha, String
    titulo, String autor2,
024             String titulo2, String lpublicacion, String
    publicacion, String lugarimpresi,
025             String edicion, String autorsub, String isbn, String
    nota, String signatura, String congreso, String file,
026             String materia, String resumen) {
027 
028         this.autor = autor;
029         this.fecha = fecha;
030         this.titulo = titulo;
031         this.autor2 = autor2;
032         this.titulo2 = titulo2;
033         this.lpublicacion = lpublicacion;
034         this.publicacion = publicacion;
035         this.lugarimpresi = lugarimpresi;
036         this.edicion = edicion;
037         this.autorsub = autorsub;
038         this.isbn = isbn;
039         this.nota = nota;
040         this.signatura = signatura;
041         this.congreso = congreso;
042         this.file = file;
043         this.materia = materia;
044         this.resumen = resumen;
```

```
045     }
046
047     public String getAutor() {
048         return autor;
049     }
050
051     public void setAutor(String autor) {
052         this.autor = autor;
053     }
054
055     public String getFecha() {
056         return fecha;
057     }
058
059     public void setFecha(String fecha) {
060         this.fecha = fecha;
061     }
062
063     public String getTitulo() {
064         return titulo;
065     }
066
067     public void setTitulo(String titulo) {
068         this.titulo = titulo;
069     }
070
071     public String getAutor2() {
072         return autor2;
073     }
074
075     public void setAutor2(String autor2) {
076         this.autor2 = autor2;
077     }
078
079     public String getTitulo2() {
080         return titulo2;
081     }
082
083     public void setTitulo2(String titulo2) {
084         this.titulo2 = titulo2;
085     }
086
087     public String getLpublicacion() {
088         return lpublicacion;
089     }
090
091     public void setLpublicacion(String lpublicacion) {
092         this.lpublicacion = lpublicacion;
093     }
094
095     public String getPublicacion() {
096         return publicacion;
097     }
098
099     public void setPublicacion(String publicacion) {
100         this.publicacion = publicacion;
```

```
101     }
102
103     public String getLugarimpresi() {
104         return lugarimpresi;
105     }
106
107     public void setLugarimpresi(String lugarimpresi) {
108         this.lugarimpresi = lugarimpresi;
109     }
110
111     public String getEdicion() {
112         return edicion;
113     }
114
115     public void setEdicion(String edicion) {
116         this.edicion = edicion;
117     }
118
119     public String getAutorsub() {
120         return autorsub;
121     }
122
123     public void setAutorsub(String autorsub) {
124         this.autorsub = autorsub;
125     }
126
127     public String getIsbn() {
128         return isbn;
129     }
130
131     public void setIsbn(String isbn) {
132         this.isbn = isbn;
133     }
134
135     public String getNota() {
136         return nota;
137     }
138
139     public void setNota(String nota) {
140         this.nota = nota;
141     }
142
143     public String getSignatura() {
144         return signatura;
145     }
146
147     public void setSignatura(String signatura) {
148         this.signatura = signatura;
149     }
150
151     public String getCongreso() {
152         return congreso;
153     }
154
155     public void setCongreso(String congreso) {
156         this.congreso = congreso;
```

```
157         }
158
159         public String getFile() {
160             return file;
161         }
162
163         public void setFile(String file) {
164             this.file = file;
165         }
166
167         public String getMateria() {
168             return materia;
169         }
170
171         public void setMateria(String materia) {
172             this.materia = materia;
173         }
174
175         public String getResumen() {
176             return resumen;
177         }
178
179         public void setResumen(String resumen) {
180             this.resumen = resumen;
181         }
182 }
```

Tabla 13. Clase con los campos de Endnote-Refworks

E. DATA CURATION

Todos los cambios que se realizaron en el proceso de Data Curation o Limpieza de Datos, por columna, fueron los siguientes:

Cabecera (Columna 000)

Debido a que todos los caracteres ocupan posiciones fijas me quedé con los tres valores que me interesaban (Estado del registro, Tipo de registro y Nivel bibliográfico), los cuales se encuentran en las posiciones 5, 6 y 7, respectivamente. Para ello se ejecutó a través de la opción "Transformar" la expresión: value.replace(/(^.....)(...)(.*)/ , "$2") (Ilustración 21)

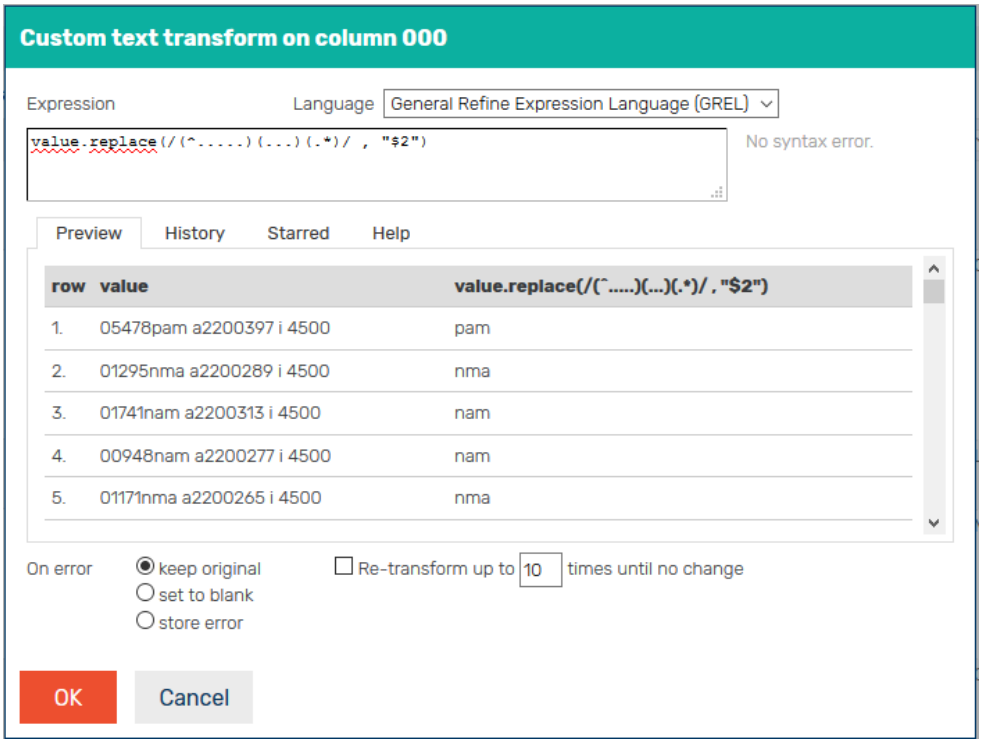

Ilustración 21. Transformación de la columna 000 en GraphDB

Como resultado se obtuvo una columna con tres caracteres y tras ello se realizó, a través de la opción "Separar en varias columnas" una división de esta en tres columnas por longitud, seleccionando cada una de ellas recogiese uno de los caracteres (1,1,1) (Ilustración 22).

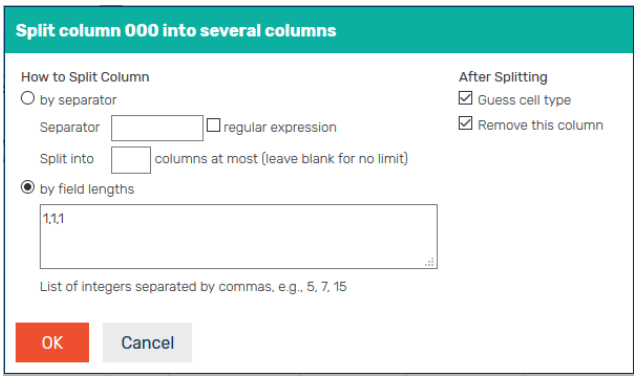

Ilustración 22. Separación de la columna 000 en varias columnas en GraphDB

El siguiente paso fue modificar cada una de ellas individualmente:

- Para la primera de ellas se cambió el nombre de columna a "E-Registro" y a través de la búsqueda facetada se renombraron las tres letras que hacen mención al estado del registro de la siguiente manera (Ilustración 23):
  - c → Corregido
  - n → Nuevo
  - p → Aumentado
- En la siguiente se hicieron los mismos cambios, se renombró a "T-Registro" y se cambiaron los valores de este modo:
  - a → Textual
  - g → Material gráfico proyectable
  - k → Material gráfico bidimensional
  - m → Archivo de ordenador
- Por último, el nombre que se puso fue "N-Biblio" y se renombraron:
  - a y m → mono
  - s → serl

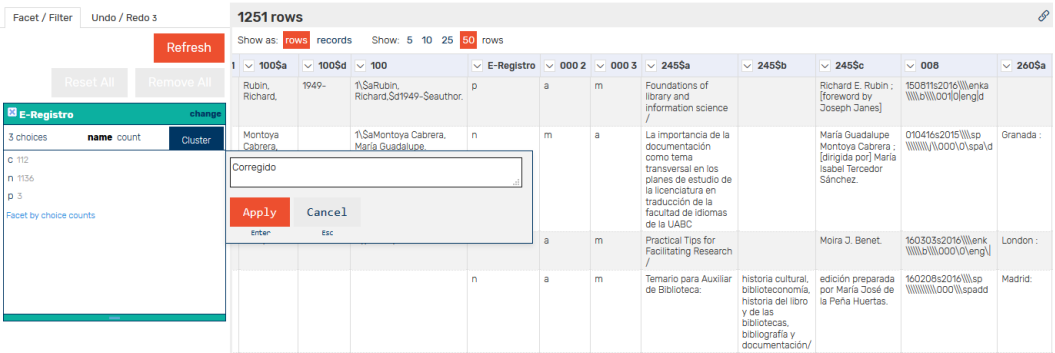

Ilustración 23. Modificación de uno de los valores de la columna "E-Registro" en GraphDB

Campo 008 (Columna 008)

Del mismo modo que antes, realice en primer lugar una transformación de dicha columna para extraer de ella el idioma, por ser también valores fijos, ejecutando en GREL la siguiente expresión: value.replace(/(^.....)(...)(.*)/,"$2")

De este modo, la columna 008 quedo reducida únicamente a los códigos de lengua[32], que fueron revisados encontrándose varios de ellos erróneos que fueron renombrados de la siguiente manera para que tuviesen el código correcto, previa revisión del registro para asegurar que se trataba efectivamente de ese:

- sp → spa
- esp → spa (no son registros en esperanto)
- gao → glg
- ne → mul
- ag → mul

Subcampo 100$a (Columna 100$a)

Los datos de este campo se encuentran todos normalizados, separando los apellidos del nombre por medio de una coma, y la mayoría de ellos con un espacio, coma y/o punto al final.

Lo que se hizo en este caso fue obtener dos columnas, una con los nombres de los autores normalizados perfectamente y otra sin normalizar. En consecuencia, en primer lugar se corrigieron los datos:

- Se eliminaron los puntos al final del nombre, salvo en casos en que fuese de una inicial. Para ello se lanzó esta expresión: value.replace(/([a-záéíóú])(\.)/ , "$1")
- Se eliminaron las comas al final de los nombres, no las del medio que separan apellidos y nombre, ejecutando la expresión: value.replace(/,$/ , "")
- Los nombres se unificaron mediante la agrupación en clusters, a través de la opción “Cluster & editar” (Ilustración 24) para unificar los nombres.
- Así mismo, y de cara a usar dichos nombres como parte de las URI, se hizo una columna basada en esta, a través de “Agregar columna basada en esta columna”, con la expresión “fingerprint(value)” para suprimir aquellos caracteres especiales, como los acentos, que no pueden incluirse en una dirección URL.

[32] https://www.loc.gov/marc/languages/language_code.html

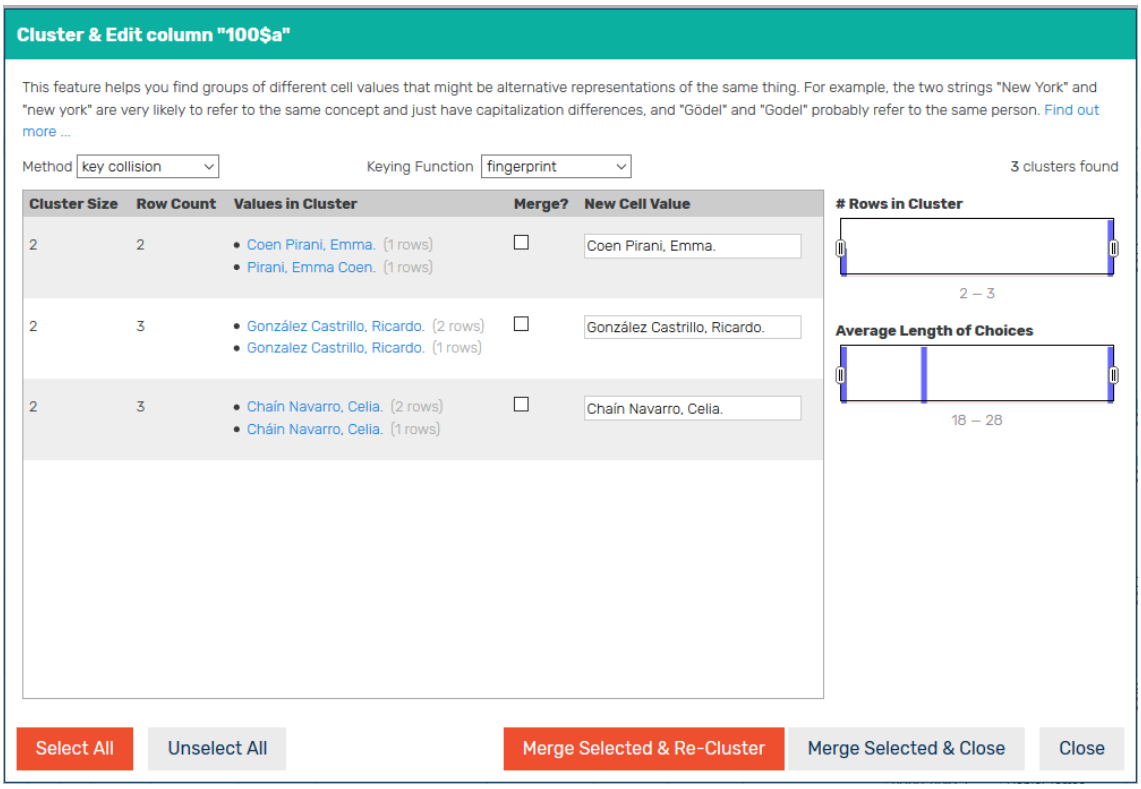

Ilustración 24. Agrupación por clusters de nombres de autores en GraphDB

El siguiente paso fue el de "desnormalizar" los nombres, para lo cual se realizó una copia de la columna con el nombre "Autor" y se dividió en dos ("Autor 1" con el apellido y "Autor 2" con el nombre), indicando que la expresión regula ",\s" fuese el separador y luego se unieron añadiendo una nueva columna, titulada "Autor-completo", en la que se ejecutó la expresión: cells["Autor 2"].value + cells["Autor 1"].value

Subcampo 100$d (Columna 100$d)

En esta columna el único cambio que se realizó fue el de eliminar los puntos que aparecían al final de algunas de las fechas a través de la expresión: value.replace(/\.$/,"")

Subcampo 245$a (Columna 245$a)

Para este primer subcampo es posible que el título apareciese con una diagonal (/) al final, lo que indica que el siguiente subcampo estará vacío, pues pasaría a las menciones de responsabilidad. En el caso contrario, lo que aparecería al final podría ser dos puntos (:), un igual (=) o un punto (.). Por ello, lo que se hizo fue establecer en una columna el título principal:

- Se agregó una columna basada en la 245 (la cual incluye los tres subcampos $a, $b y $c) llamada “Titulo-principal”
- Se eliminaron las diagonales, puntos, dos puntos e iguales al final de los dos campos, considerando que en algunos aparece sin un espacio en blanco entre ella y el último carácter de la cadena, ejecutando dicha transformación por medio de la expresión: value.replace(/(\s\/$)|(\/$)|(\s\:$)|(\:$)|(\s\.$)|(\.$)|(\s\=$)|(\=$)/ , "")

Subcampo 245$b (Columna 245$b)

Para este se agregó una columna basada en esta llamada “Subtitulo”, y se eliminaron de ella las diagonales y puntos que pudiesen aparecer al final con la expresión: value.replace(/(\s\/$)|(\/$)|(\s\.$)/ , "")

Subcampo 245$c (Columna 245$c)

En este, la única modificación que se llevó a cabo fue la de eliminar los posibles puntos finales realizando esta transformación: value.replace(/(\.$)/ , "")

Subcampo 260$a (Columna 260$a)

En primer lugar, se agregó una columna basada en esta con el título “Lugar”. Tras ello se eliminaron los siguientes elementos en esta nueva columna a través de transformaciones con diferentes expresiones:

- Dos puntos al final: value.replace(/(\s$)|(\s\:$)|(\:$)/,"")
- Espacios en blanco al final: value.replace(/\s\:/,"")
- Los “[etc.]”: value.replace(/(\s\[etc\.\])|(\[etc\.\])|(\s\[etc\])/ , "")
- Los corchetes (a través de dos expresiones):
    - value.replace(/(\))|(^\()|(^\[)|(\])/ , "")
    - value.replace(/(\s\[)|(\s\()|(\s\:\;)|(\s\;\;)/,", ")

Una vez hecho eso, y del mismo modo que con los autores, aunque en esta ocasión con una mayor cantidad de elementos (Ilustración 25), se agruparon y unieron los términos con los clusters.

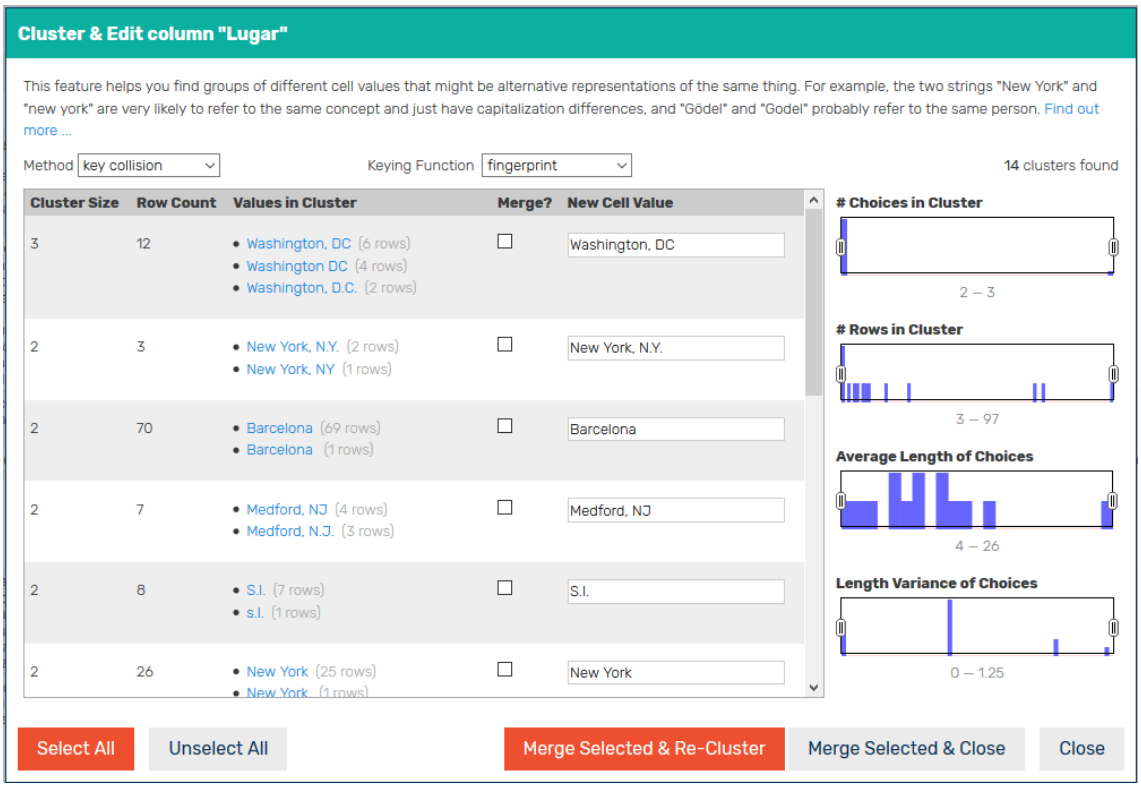

Ilustración 25. Agrupación por clusters de nombres lugares de publicación en GraphDB

Subcampo 260$b (Columna 260$b)

Los cambios llevados a cabo, así como el estado de los datos, son muy similares al del caso anterior. Se renombró la columna a "Editor" y se eliminaron:

- Espacios en blanco al principio y final: value.replace(/(\s$)|(^\s)/ , "")
- Puntos y comas al final, considerando los espacios en blanco: value.replace(/(\s\.$)|(\.$)|(\s,$)|(,$)/ , "")
- Los "[etc.]": value.replace(/(\s\[etc\.\])|(\[etc\.\])|(\s\[etc\])/ , "")
- Los corchetes: value.replace(/(\[)|(\]$)/ , "")

Tras ello, se volvió a hacer una agrupación por clusters unificando con ello los nombres de algunas editoriales.

Subcampo 260$c (Columna 260$c)

En primer lugar se agregó una columna titulada "Fecha" y tras ello se suprimieron los siguientes elementos:

- Espacios en blanco al principio y final: value.replace(/(^\s)(\s$)/,"")
- Puntos y comas al final: value.replace(/(\s\.$)|(\.$)|(\s,$)|(,$)/ , "")

- Corchetes: value.replace(/(\[)|(\])/ , "")
- Alusiones al copyright: value.replace(/(cop\.\s)|(cop\.)|(cop)|(c)/ , "")
- Alusiones al depósito legal: value.replace(/(D\.L\.\s)|(DL\s)/ , "")

Campo 650 (Columna 650)

En este caso se partió directamente de la columna 650, que incluye tanto el subcampo $a como el $x de cara a poder establecer una mejor relación precisamente entre ambos subcampos, pudiendo tratarlos como dos términos independientes y como un uno solo formado por un término principal con una subdivisión de materia. Ya que si extraigo directamente los valores ambos subcampos puede darse el caso en el que en un registro aparezcan dos subcampos $a y uno $x, siendo imposible saber cuál de los dos es el término principal.

Para ello lanzó una búsqueda normal en la que se buscaron todos los registros que incluyen un subcampo $x y se agregó una columna basada en ellos llamada “Complejos” con el mismo contenido.

Tras ello, y partiendo otra vez de la columna 650, se añadió otra columna basada en ella llamada “Simples” destinada a incluir todas las materias principales que no tuviesen ninguna subdivisión, excluyéndose así los que tienen $x, con la expresión: if(value.contains("$x"), "", value)

Acto seguido se lanzó una búsqueda facetada en “Complejos” para seleccionar manualmente las materias que careciesen sin subdivisión, ya que como se comentó el problema se encuentra precisamente en estos, y los traspasé a “Simples”. De este modo, en “Complejos” se encuentran únicamente términos principales seguidos de su subdivisión.

Después elimino de “Simples” y “Complejos el código de subcampo, $a, y los puntos con “value.replace(/(\\4\$a)|(\.)/,"")” y sustituyo el subcampo $x de “Complejos” por dos guiones con: value.replace(/\$x/,"--")

Por último, separo ambas columnas en varias usando como separador el punto y coma, y además las cuatro subdivisiones de “Complejos” las vuelvo a dividir usando como separador los dos guiones, y sin eliminar la columna de origen.

ISBN (Columna 020$a)

Por último, para los códigos ISBN se han eliminado:

- Los espacios entre ISBN y paréntesis: value.replace(/(\s\()/ , "(")
- Los espacios y dos puntos al final: value.replace(/(\s:\s$)|(\s:$)|(\s$)|(:$)/ , "")
- Los espacios después de puntos: value.replace(/(\.\s)/ , ".")
- Las cadenas t 1, t 2 y los espacios tras puntos para hacer mejor la división: value.replace(/(t\s[0-9])|(\.\s)/, "")
- Otros espacios en blanco dentro de paréntesis: value.replace(/(\s:\s)|(:\s)|(:)|([a-z]\s[a-z])|([a-z]\s[A-Z])|([a-z]\s[0-9])/, "")

Asimismo, manualmente se realizaron modificaciones y nuevas agrupaciones por clusters.

F. ESTUDIO DE LOS DATOS DE LA BNE

Los procesos descritos anteriormente para las dos primeras etapas fueron realizados con datos de la Biblioteca Nacional de España. En esta ocasión con el objetivo de comprobar si es posible optar aquí por el camino más rápido y adecuado y observar las diferencias respecto a la Biblioteca de la Universidad de Granada.

Identificación de los datos

Debido a que el formato idóneo es el MARC, a través de MarcEdit 6 se hizo la consulta para descargar desde el servidor Z39.50 los registros necesarios en dicho formato.

Antes de lanzarla, se tuvo que modificar el número de registros a recuperar a 1.251, para tener una muestra igual a la anterior. También, hubo que cambiar el tipo de búsqueda que realiza el campo "Subject", estableciendo la opción "Phrase". De este modo se podrán usar booleanos para ejecutar exactamente la misma búsqueda.

Una vez hecho, la consulta lanzada fue: "documentación OR biblioteconomía". Se descargaron todos los registros rápidamente en formato ISO 2709. Tras ello, a través de la herramienta de exportación de registros delimitados por tabuladores, se cargaron las opciones de exportación que usadas con anterioridad y exportaron exactamente los mismos campos y subcampos delimitados por comas.

Este paso, el de descargar los registros y extraer los datos, ha sido mucho más rápido, sin existir ningún inconveniente o limitación para ello.

Identificación de la licencia

A través de página web, hay un enlace en el pie de página "Aviso legal"[33]. Ahí aparece la información con todo detalle: "Los datos del Catálogo se encuentran bajo la licencia CC0, por lo que su uso es gratuito y no requiere autorización previa. En cualquier caso, la Biblioteca agradecerá la mención del origen de los registros".

Limpieza de los datos

La limpieza de los datos se realizó del mismo modo, prestando atención a cada columna/subcampo para localizar primero sus errores más comunes y solucionarlos de manera masiva y luego prestar atención a problemas menores. Para ello la búsqueda facetada volvió a servir de gran ayuda.

[33] http://www.bne.es/es/NavegacionRecursiva/Pie/avisoLegal/

Subcampo 100$a

Este subcampo está perfectamente recogido, no se encuentran los errores de puntos (el error más importante de la BUG al confundirse con el de las iniciales de cara al querer eliminarlo), comas y espacios detrás de ellos.

De este modo, para desnormalizar el nombre solo habría que dividir la columna en dos en base al punto y reagruparlas.

Subcampos 245$a y 245$b

Tampoco encontramos espacios o diagonales al final del dichos subcampos. Salvo en un campo, los signos de igual (=) y dos puntos (:) se encuentran separados por un espacio del título. El punto si está unido a él.

Es por ello que solo habría que modificar los puntos, ya que salvo en un caso son usados de cara a introducir después el subcampo $p del nombre de parte o sección.

Una vez hecho ya podríamos unir las columnas y separar los títulos paralelos.

Subcampo 260$a

No se encuentran espacios en blanco ni dos puntos al final. No obstante, a la limpieza de corchetes hay que sumar la del control de autoridades, ya que tampoco está todo perfectamente normalizado. Por ejemplo, para una misma ubicación se encuentran hasta cinco variantes con cambios mínimos entre ellas:

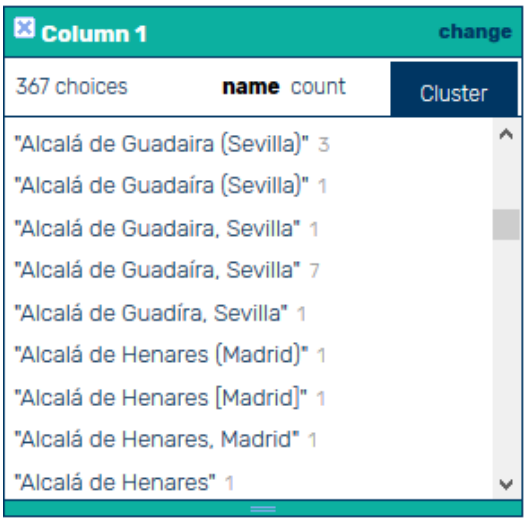

Ilustración 26. Búsqueda facetada del subcampo 260$a de la BNE en GraphDB

Subcampo 260$b

Este campo peca de los mismos aciertos y errores que el anterior.

Subcampo 260$c

Igual que lo anteriores, hay fallos al hacer referencia al D.L. en dos casos y copyright.

Subcampo 650$a

No se hacen usos de puntos, por lo que no hay que borrar ningúno, pero al mismo tiempo se dificulta la división en varias columnas, ya que no es posible usar los espacios con los que separa las palabras de un mismo término de otro diferente. Es por ello que habría que buscar un camino alternativo o modificar la extracción de los datos para poder obtener todos los datos de este subcampo por separado.

Subcampo 650$x

Este campo ofrece datos mucho más variados. Igual que el anterior no ofrece errores de puntos, comas o espacios que haya que eliminar para limpiarlos. No obstante, ofrece exactamente el mismo gran problema.

Subcampo 655$a

En este caso no hay ningún problema, se puede hacer directamente la división sin tener que hacer nada más antes.

Subcampo 020$a

Todos los códigos ISBN están perfectamente establecidos y, al no introducir ningún texto o especificación, se puede realizar la división a través de los espacios que deja entre los códigos sin ningún problema.

Subcampo 080$a

Del mismo modo que con la Biblioteca de la Universidad de Granada, la aparición de nombres dificulta la separación en varias columnas. En el caso anterior estos casos eran mínimo, ahora no lo son tanto, hay un total de 121 registros con nombres, por lo que la solución será distinta.

G. GENERACIÓN DE RDF EN BIBFRAME 2.0

En primer lugar, se hizo una amplia consulta para construir de manera general las principales clases (Tabla 14) y ya luego por separado ir profundizando por medio de otras en cada una de ellas. De este modo se evita el generar grafos vacios.

```
PREFIX bf: <http://id.loc.gov/ontologies/bibframe/>
PREFIX bflc: <http://id.loc.gov/ontologies/bflc/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix madsrdf: <http://www.loc.gov/mads/rdf/v1#>
prefix spif: <http://spinrdf.org/spif#>

INSERT{ GRAPH<http://example.org/>{

    ?autorURI a bf:Agent, bf:Person ;
        rdfs:label ?autor;
        bflc:name00MarcKey ?MarcKey;
        bflc:name00MatchKey ?autor;
        bflc:primaryContributorName00MatchKey ?autor.

    ?workURI a ?tipoR, bf:Work;
        rdfs:label ?titulo;
        bf:language ?idiomaURI;
        bf:adminMetadata [ a bf:AdminMetadata ;
        bf:status [ a bf:Status ;
        bf:code ?estadoR ]];
        bf:hasInstance ?instanceURI;
        bf:subject ?topicS1URI;
        bf:subject ?topicS2URI;
        bf:subject ?topicS3URI;
        bf:subject ?topicS4URI;
        bf:subject ?topicS5URI;
        bf:subject ?topicS6URI;
        bf:subject ?topicS7URI;
        bf:subject ?topicS8URI;
        bf:subject ?topicS9URI;
        bf:subject ?topicS10URI;
        bf:subject ?topicS11URI;
        bf:subject ?topicS12URI;
        bf:subject ?topicS13URI;
        bf:subject ?topicS14URI;
        bf:subject ?topicS15URI;
        bf:subject ?topicC1URI;
        bf:subject ?topicC2URI;
        bf:subject ?topicC3URI;
        bf:subject ?topicC4URI.

    ?instanceURI a bf:Instance, ?electronico ;
        rdfs:label ?titulo;
        bf:instanceOf ?workURI;
        bf:issuance ?nivelURI;
        bf:provisionActivityStatement ?Publicacion ;
```

```
        bf:responsibilityStatement ?tituloc .

    ?idiomaURI a bf:Language.

    ?nivelURI a bf:Issuance.

    ?topicS1URI a bf:Topic, madsrdf:Topic ;
         rdfs:label ?simple1.
    ?topicS2URI a bf:Topic, madsrdf:Topic ;
        rdfs:label ?simple2.
    ?topicS3URI a bf:Topic, madsrdf:Topic ;
        rdfs:label ?simple3.
    ?topicS4URI a bf:Topic, madsrdf:Topic ;
        rdfs:label ?simple4.
    ?topicS5URI a bf:Topic, madsrdf:Topic ;
        rdfs:label ?simple5.
    ?topicS6URI a bf:Topic, madsrdf:Topic ;
        rdfs:label ?simple6.
    ?topicS7URI a bf:Topic, madsrdf:Topic ;
        rdfs:label ?simple7.
    ?topicS8URI a bf:Topic, madsrdf:Topic ;
        rdfs:label ?simple8.
    ?topicS9URI a bf:Topic, madsrdf:Topic ;
        rdfs:label ?simple9.
    ?topicS10URI a bf:Topic, madsrdf:Topic ;
        rdfs:label ?simple10.
    ?topicS11URI a bf:Topic, madsrdf:Topic ;
        rdfs:label ?simple11.
    ?topicS12URI a bf:Topic, madsrdf:Topic ;
        rdfs:label ?simple12.
    ?topicS13URI a bf:Topic, madsrdf:Topic ;
        rdfs:label ?simple13.
    ?topicS14URI a bf:Topic, madsrdf:Topic ;
        rdfs:label ?simple14.
    ?topicS15URI a bf:Topic, madsrdf:Topic ;
        rdfs:label ?simple15.

    ?topicC1URI a bf:Topic, madsrdf:ComplexSubject ;
        rdfs:label ?complejo1.
    ?topicC2URI a bf:Topic, madsrdf:ComplexSubject ;
        rdfs:label ?complejo2.
    ?topicC3URI a bf:Topic, madsrdf:ComplexSubject ;
        rdfs:label ?complejo3.
    ?topicC4URI a bf:Topic, madsrdf:ComplexSubject ;
        rdfs:label ?complejo4.
    }
        }
WHERE {
    service <http://localhost:7200/rdf-bridge/1669481892565> {
        ?registroRow a <urn:Row> ;
                    <urn:col:001> ?numero ;
                    <urn:col:E-Registro> ?eRegistro ;
                    <urn:col:T-Registro> ?tRegistro ;
                    <urn:col:N-Biblio> ?nBiblio ;

                     VALUES (?eRegistro ?estadoR) {
```

```
104           ("Corregido" "c" )
105           ("Nuevo" "n")
106           ("Aumentado" "p")}
107 
108                         VALUES (?tRegistro ?tipoR) {
109           ("Textual" bf:Text )
110           ("Archivo de ordenador" UNDEF )
111           ("Material gráfico proyectable" bf:MovingImage)
112           ("Material gráfico bidimensional" bf:StillImage)}
113 
114                         VALUES (?tRegistro ?electronico) {
115           ("Textual" UNDEF )
116           ("Archivo de ordenador" bf:Electronic )
117           ("Material gráfico proyectable" UNDEF)
118           ("Material gráfico bidimensional" UNDEF)}
119 
120         OPTIONAL{?registroRow <urn:col:008> ?idioma ;}
121         OPTIONAL{?registroRow <urn:col:100> ?autorIndicadores ;}
122         OPTIONAL{?registroRow <urn:col:100-a> ?autor ;}
123         OPTIONAL{?registroRow <urn:col:100-d> ?fechaAutor ;}
124         OPTIONAL{?registroRow <urn:col:Autor-URI> ?autoruri ;}
125         OPTIONAL{?registroRow <urn:col:245-a> ?tituloa ;}
126         OPTIONAL{?registroRow <urn:col:245-b> ?titulob ;}
127         OPTIONAL{?registroRow <urn:col:245-c> ?tituloc ;}
128         OPTIONAL{?registroRow <urn:col:260-a> ?publicaciona ;}
129         OPTIONAL{?registroRow <urn:col:260-b> ?publicacionb ;}
130         OPTIONAL{?registroRow <urn:col:260-c> ?publicacionc ;}
131         OPTIONAL{?registroRow <urn:col:Simple1> ?simple1 ;}
132         OPTIONAL{?registroRow <urn:col:Simple2> ?simple2 ;}
133         OPTIONAL{?registroRow <urn:col:Simple3> ?simple3 ;}
134         OPTIONAL{?registroRow <urn:col:Simple4> ?simple4 ;}
135         OPTIONAL{?registroRow <urn:col:Simple5> ?simple5 ;}
136         OPTIONAL{?registroRow <urn:col:Simple6> ?simple6 ;}
137         OPTIONAL{?registroRow <urn:col:Simple7> ?simple7 ;}
138         OPTIONAL{?registroRow <urn:col:Simple8> ?simple8 ;}
139         OPTIONAL{?registroRow <urn:col:Simple9> ?simple9 ;}
140         OPTIONAL{?registroRow <urn:col:Simple10> ?simple10 ;}
141         OPTIONAL{?registroRow <urn:col:Simple11> ?simple11 ;}
142         OPTIONAL{?registroRow <urn:col:Simple12> ?simple12 ;}
143         OPTIONAL{?registroRow <urn:col:Simple13> ?simple13 ;}
144         OPTIONAL{?registroRow <urn:col:Simple14> ?simple14 ;}
145         OPTIONAL{?registroRow <urn:col:Simple15> ?simple15 ;}
146         OPTIONAL{?registroRow <urn:col:Simple1-URI> ?simple1URI ;}
147         OPTIONAL{?registroRow <urn:col:Simple2-URI> ?simple2URI ;}
148         OPTIONAL{?registroRow <urn:col:Simple3-URI> ?simple3URI ;}
149         OPTIONAL{?registroRow <urn:col:Simple4-URI> ?simple4URI ;}
150         OPTIONAL{?registroRow <urn:col:Simple5-URI> ?simple5URI ;}
151         OPTIONAL{?registroRow <urn:col:Simple6-URI> ?simple6URI ;}
152         OPTIONAL{?registroRow <urn:col:Simple7-URI> ?simple7URI ;}
153         OPTIONAL{?registroRow <urn:col:Simple8-URI> ?simple8URI ;}
154         OPTIONAL{?registroRow <urn:col:Simple9-URI> ?simple9URI ;}
            OPTIONAL{?registroRow <urn:col:Simple10-URI> ?simple10URI
155 ;}
            OPTIONAL{?registroRow <urn:col:Simple11-URI> ?simple11URI
156 ;}
157         OPTIONAL{?registroRow <urn:col:Simple12-URI> ?simple12URI
```

```
;}
        OPTIONAL{?registroRow <urn:col:Simple13-URI> ?simple13URI
;}
        OPTIONAL{?registroRow <urn:col:Simple14-URI> ?simple14URI
;}
        OPTIONAL{?registroRow <urn:col:Simple15-URI> ?simple15URI
;}

        OPTIONAL{?registroRow <urn:col:Complejos1> ?complejo1 ;}
        OPTIONAL{?registroRow <urn:col:Complejos2> ?complejo2 ;}
        OPTIONAL{?registroRow <urn:col:Complejos3> ?complejo3 ;}
        OPTIONAL{?registroRow <urn:col:Complejos4> ?complejo4 ;}


        OPTIONAL{?registroRow <urn:col:Complejos1-URI>
?complejo1URI ;}
        OPTIONAL{?registroRow <urn:col:Complejos2-URI>
?complejo2URI ;}
        OPTIONAL{?registroRow <urn:col:Complejos3-URI>
?complejo3URI ;}
        OPTIONAL{?registroRow <urn:col:Complejos4-URI>
?complejo4URI ;}


        bind(iri(concat("http://example.org/",
spif:encodeURL(?autoruri))) as ?autorURI)
        bind(concat("100",?autorIndicadores," $a",?autor ) as
?MarcKey)
        bind(concat("100",?autorIndicadores," $a",?autor,"
$d",?fechaAutor ) as ?MarcKey)

        bind(concat(?publicaciona, ?publicacionb, ?publicacionc)
as ?Publicacion)

        bind(concat(?tituloa, ?titulob) as ?titulo)

        bind(iri(concat("http://example.org/", ?numero, "#Work"))
as ?workURI)
        bind(iri(concat("http://example.org/", ?numero,
"#Instance")) as ?instanceURI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?simple1URI)) as ?topicS1URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?simple2URI)) as ?topicS2URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?simple3URI)) as ?topicS3URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?simple4URI)) as ?topicS4URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?simple5URI)) as ?topicS5URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?simple6URI)) as ?topicS6URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?simple7URI)) as ?topicS7URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?simple8URI)) as ?topicS8URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?simple9URI)) as ?topicS9URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
```

```
?simple10URI)) as ?topicS10URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?simple11URI)) as ?topicS11URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?simple12URI)) as ?topicS12URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?simple13URI)) as ?topicS13URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?simple14URI)) as ?topicS14URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?simple15URI)) as ?topicS15URI)

        bind(iri(concat("http://example.org/", "#Topic650-",
?complejo1URI)) as ?topicC1URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?complejo2URI)) as ?topicC2URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?complejo3URI)) as ?topicC3URI)
        bind(iri(concat("http://example.org/", "#Topic650-",
?complejo4URI)) as ?topicC4URI)


        bind(iri(concat("http://id.loc.gov/vocabulary/languages/",
?idioma)) as ?idiomaURI)
        bind(iri(concat("http://id.loc.gov/vocabulary/issuance/",
?nBiblio)) as ?nivelURI)

        }
}
```

Tabla 14. Creación de las principales clases de BIBFRAME en SPARQL

Tras ello se vincularon las obras a sus autores (Tabla 15).

```
PREFIX bf: <http://id.loc.gov/ontologies/bibframe/>
PREFIX bflc: <http://id.loc.gov/ontologies/bflc/>
prefix spif: <http://spinrdf.org/spif#>


INSERT{ GRAPH<http://example.org/>{

    ?workURI a ?tipoR, bf:Work;
        bf:contribution [ a bflc:PrimaryContribution;
        bf:agent ?autorURI] ;

    }
        }
WHERE {
    service <http://localhost:7200/rdf-bridge/1669481892565> {
        ?registroRow a <urn:Row> ;
        <urn:col:001> ?numero ;
        <urn:col:T-Registro> ?tRegistro ;
        <urn:col:Autor-URI> ?autoruri ;


                    VALUES (?tRegistro ?tipoR) {
        ("Textual" bf:Text )
```

```
        ("Archivo de ordenador" UNDEF )
        ("Material gráfico proyectable" bf:MovingImage)
        ("Material gráfico bidimensional" bf:StillImage)}

        bind(iri(concat("http://example.org/",
spif:encodeURL(?autoruri))) as ?autorURI)
        bind(iri(concat("http://example.org/", ?numero, "#Work"))
as ?workURI)
        }
}
```

Tabla 15. Vinculación entre obras y autores en SPARQL

Después se ejecutaron cuatro consultas muy similares para las cuatro columnas de encabezamientos de materias compuestas por un encabezamiento principal y una subdivisión de materia general (Tabla 16;Tabla 17;Tabla 18;Tabla 19).

```
PREFIX bf: <http://id.loc.gov/ontologies/bibframe/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix madsrdf: <http://www.loc.gov/mads/rdf/v1#>
prefix spif: <http://spinrdf.org/spif#>


INSERT{ GRAPH<http://example.org/>{
    ?topicCURI a bf:Topic, madsrdf:ComplexSubject ;
        rdfs:label ?complejo;
        madsrdf:componentList ( [ a madsrdf:Topic ;
                madsrdf:authoritativeLabel ?complejo1 ] [ a
madsrdf:GenreForm ;
                madsrdf:authoritativeLabel ?complejo2 ] ) .
    }
        }
WHERE {
    service <http://localhost:7200/rdf-bridge/1669481892565> {
        ?registroRow a <urn:Row> ;
        <urn:col:Complejos1> ?complejo ;
        <urn:col:Complejos11> ?complejo1 ;
        <urn:col:Complejos12> ?complejo2 ;
        <urn:col:Complejos1-URI> ?complejoURI ;

        bind(iri(concat("http://example.org/", "#Topic650-",
?complejoURI)) as ?topicCURI)
        }
}
```

Tabla 16. SPARQL para añadir la primera columna de encabezamientos de materia con subdivisión de materia general

```
PREFIX bf: <http://id.loc.gov/ontologies/bibframe/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix madsrdf: <http://www.loc.gov/mads/rdf/v1#>
prefix spif: <http://spinrdf.org/spif#>

```

```
INSERT{ GRAPH<http://example.org/>{
    ?topicCURI a bf:Topic, madsrdf:ComplexSubject ;
        rdfs:label ?complejo;
        madsrdf:componentList ( [ a madsrdf:Topic ;
                madsrdf:authoritativeLabel ?complejo1 ] [ a
madsrdf:GenreForm ;
                madsrdf:authoritativeLabel ?complejo2 ] ) .
    }
        }
WHERE {
    service <http://localhost:7200/rdf-bridge/1669481892565> {
        ?registroRow a <urn:Row> ;
        <urn:col:Complejos2> ?complejo ;
        <urn:col:Complejos21> ?complejo1 ;
        <urn:col:Complejos22> ?complejo2 ;
        <urn:col:Complejos2-URI> ?complejoURI ;

        bind(iri(concat("http://example.org/", "#Topic650-",
?complejoURI)) as ?topicCURI)
        }
}
```

Tabla 17. SPARQL para añadir la segundaa columna de encabezamientos de materia con subdivisión de materia general

```
PREFIX bf: <http://id.loc.gov/ontologies/bibframe/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix madsrdf: <http://www.loc.gov/mads/rdf/v1#>
prefix spif: <http://spinrdf.org/spif#>


INSERT{ GRAPH<http://example.org/>{
    ?topicCURI a bf:Topic, madsrdf:ComplexSubject ;
        rdfs:label ?complejo;
        madsrdf:componentList ( [ a madsrdf:Topic ;
                madsrdf:authoritativeLabel ?complejo1 ] [ a
madsrdf:GenreForm ;
                madsrdf:authoritativeLabel ?complejo2 ] ) .
    }
        }
WHERE {
    service <http://localhost:7200/rdf-bridge/1669481892565> {
        ?registroRow a <urn:Row> ;
        <urn:col:Complejos3> ?complejo ;
        <urn:col:Complejos31> ?complejo1 ;
        <urn:col:Complejos32> ?complejo2 ;
        <urn:col:Complejos3-URI> ?complejoURI ;

        bind(iri(concat("http://example.org/", "#Topic650-",
?complejoURI)) as ?topicCURI)
        }
}
```

Tabla 18. SPARQL para añadir la tercera columna de encabezamientos de materia con subdivisión de materia general

```
PREFIX bf: <http://id.loc.gov/ontologies/bibframe/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix madsrdf: <http://www.loc.gov/mads/rdf/v1#>
prefix spif: <http://spinrdf.org/spif#>


INSERT{ GRAPH<http://example.org/>{
    ?topicCURI a bf:Topic, madsrdf:ComplexSubject ;
        rdfs:label ?complejo;
        madsrdf:componentList ( [ a madsrdf:Topic ;
                madsrdf:authoritativeLabel ?complejo1 ] [ a
madsrdf:GenreForm ;
                madsrdf:authoritativeLabel ?complejo2 ] ) .
    }
        }
WHERE {
    service <http://localhost:7200/rdf-bridge/1669481892565> {
        ?registroRow a <urn:Row> ;
        <urn:col:Complejos4> ?complejo ;
        <urn:col:Complejos41> ?complejo1 ;
        <urn:col:Complejos42> ?complejo2 ;
        <urn:col:Complejos4-URI> ?complejoURI ;

        bind(iri(concat("http://example.org/", "#Topic650-",
?complejoURI)) as ?topicCURI)
        }
}
```

Tabla 19. SPARQL para añadir la cuarta columna de encabezamientos de materia con subdivisión de materia general

Tras ello se insertaron los datos relativos a su publicación (Tabla 20).

```
PREFIX bf: <http://id.loc.gov/ontologies/bibframe/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix spif: <http://spinrdf.org/spif#>


INSERT{ GRAPH<http://example.org/>{

    ?instanceURI a bf:Instance, ?electronico ;
        bf:provisionActivity [ a bf:ProvisionActivity,
                bf:Publication ;
        bf:agent [ a bf:Agent ;
        rdfs:label ?editor ];
        bf:date ?fechilla ;
        bf:place [ a bf:Place ;
        rdfs:label ?lugarPublicacion ]] ;
    }
}
WHERE {
    service <http://localhost:7200/rdf-bridge/1669481892565> {
        ?registroRow a <urn:Row> ;
        <urn:col:001> ?numero ;
        <urn:col:T-Registro> ?tRegistro ;
        <urn:col:Editor> ?editor ;
```

```
        <urn:col:Fecha> ?fechaPublicacion ;
        <urn:col:Lugar> ?lugarPublicacion ;

                        VALUES (?tRegistro ?electronico) {
            ("Textual" UNDEF )
            ("Archivo de ordenador" bf:Electronic )
            ("Material gráfico proyectable" UNDEF)
            ("Material gráfico bidimensional" UNDEF)}

        bind(iri(concat("http://example.org/", ?numero,
"#Instance")) as ?instanceURI)
        BIND(STRDT(STR(?fechaPublicacion),
<http://id.loc.gov/datatypes/edtf>) AS ?fechilla)
        }
}
```

Tabla 20. SPARQL para añadir los datos de publciación

El siguiente en insertar fueron todos los datos relativos al título, tanto en las obras como en las instancias (Tabla 21).

```
PREFIX bf: <http://id.loc.gov/ontologies/bibframe/>
PREFIX bflc: <http://id.loc.gov/ontologies/bflc/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix spif: <http://spinrdf.org/spif#>


INSERT{ GRAPH<http://example.org/>{

    ?workURI a ?tipoR, bf:Work;
        bf:title [ a bf:Title ;
            rdfs:label ?titulo ;
            bflc:titleSortKey ?titulo ;
            bf:mainTitle ?tituloPrincipal;
            bf:subitle ?subtitulo] .

    ?instanceURI a bf:Instance, ?electronico ;
        bf:title [ a bf:Title ;
            rdfs:label ?titulo ;
            bflc:titleSortKey ?titulo ;
            bf:mainTitle ?tituloPrincipal;
            bf:subitle ?subtitulo] .
    }
        }
WHERE {
    service <http://localhost:7200/rdf-bridge/1669481892565> {
        ?registroRow a <urn:Row> ;
        <urn:col:001> ?numero ;
        <urn:col:T-Registro> ?tRegistro ;
        <urn:col:245-a> ?tituloa ;
        <urn:col:245-b> ?titulob ;
        <urn:col:Titulo-principal> ?tituloPrincipal ;
        <urn:col:Subtitulo> ?subtitulo ;

                        VALUES (?tRegistro ?tipoR) {
```

```
        ("Textual" bf:Text )
        ("Archivo de ordenador" UNDEF )
        ("Material gráfico proyectable" bf:MovingImage)
        ("Material gráfico bidimensional" bf:StillImage)}

                    VALUES (?tRegistro ?electronico) {
        ("Textual" UNDEF )
        ("Archivo de ordenador" bf:Electronic )
        ("Material gráfico proyectable" UNDEF)
        ("Material gráfico bidimensional" UNDEF)}

        bind(concat(?tituloa, ?titulob) as ?titulo)

        bind(iri(concat("http://example.org/", ?numero, "#Work"))
as ?workURI)
        bind(iri(concat("http://example.org/", ?numero,
"#Instance")) as ?instanceURI)
        }
}
```

Tabla 21. SPARQL para añadir los títulos a las obras e instancias

Por ultimo se añadió a instancia su primer código ISBN (Tabla 22).

```
PREFIX bf: <http://id.loc.gov/ontologies/bibframe/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix spif: <http://spinrdf.org/spif#>


INSERT{ GRAPH<http://example.org/>{

    ?instanceURI a bf:Instance, ?electronico ;
       bf:identifiedBy [ a bf:Isbn ;
            rdf:value ?isbn1 ];
    }
        }
WHERE {
    service <http://localhost:7200/rdf-bridge/1669481892565> {
        ?registroRow a <urn:Row> ;
        <urn:col:001> ?numero ;
        <urn:col:T-Registro> ?tRegistro ;
        <urn:col:ISBN1> ?isbn1;

                    VALUES (?tRegistro ?electronico) {
        ("Textual" UNDEF )
        ("Archivo de ordenador" bf:Electronic )
        ("Material gráfico proyectable" UNDEF)
        ("Material gráfico bidimensional" UNDEF)}

        bind(iri(concat("http://example.org/", ?numero,
"#Instance")) as ?instanceURI)
        }
}
```

Tabla 22. SPARQL para añadir el código ISBN a las instancias

H. ENLACE CON LEMB Y LA BNE

El primer paso es el de añadir al dump file de la BNE los nombres de los autores, recuperándolo para ello conectando a su SPARQL Endpoint (Tabla 23).

```
PREFIX esdbpp: <http://es.dbpedia.org/property/>
PREFIX esdbpr: <http://es.dbpedia.org/resource/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>
PREFIX bf: <http://id.loc.gov/ontologies/bibframe/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX bflc: <http://id.loc.gov/ontologies/bflc/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
prefix def: <http://datos.bne.es/def/>

insert{GRAPH <http://datos.bne.es/>{
        ¿s rdfs:label ¿x
    }}
WHERE{
       GRAPH <http://datos.bne.es/>{          ¿s ¿o ¿p.
}

    service <http://datos.bne.es/sparql> {
        ¿s a def:C1005;
            rdfs:label ¿x.

    }

}
```

Tabla 23. SPARQL para añadir al dump file de la BNE el nombre de los autores

Tras ello, a través de los nombres de autores, se pudo buscar los autores coincidentes entre ambos dataset y añadirles el respective enlace a VIAF (Tabla 24).

```
PREFIX esdbpp: <http://es.dbpedia.org/property/>
PREFIX esdbpr: <http://es.dbpedia.org/resource/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>
PREFIX bf: <http://id.loc.gov/ontologies/bibframe/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX bflc: <http://id.loc.gov/ontologies/bflc/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
prefix def: <http://datos.bne.es/def/>
prefix dbpedia: <http://dbpedia.org/resource/>

insert{GRAPH <http://datos.bne.es/>{
    ?a owl:sameAs ?r
    }
}
WHERE{
    ?a a bf:Person;
```

```
    rdfs:label ?autor;

                GRAPH <http://example.org/BNE>{
        ?s owl:sameAs ?p;
           rdfs:label ?autor.

    }
        service <http://datos.bne.es/sparql> {
        ?s a def:C1005;
            owl:sameAs ?r
        filter regex (?r, "^http://viaf.org/", "i")

    }
}
```

Tabla 24. SPARQL para añadir el enlace a VIAF a los autores coincidentes entre ambos datasets

Para enlazar con los materias de LEMB, se buscaron aquellas coincidentes y se les añadió la clase Concept de SKOS y los schemas a los que pertenece (Tabla 25).

```
PREFIX esdbpp: <http://es.dbpedia.org/property/>
PREFIX esdbpr: <http://es.dbpedia.org/resource/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>
PREFIX bf: <http://id.loc.gov/ontologies/bibframe/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>

INSERT { GRAPH<http://example.org/>{
    ?s a skos:Concept;
            skos:prefLabel ?x2;
            skos:inScheme
<http://id.sgcb.mcu.es/Autoridades/conceptScheme>;
            skos:inScheme
<http://id.sgcb.mcu.es/Autoridades/subjectsNames>
    }
}
WHERE{
       ?s a bf:Topic;
       rdfs:label ?x
    BIND(STRLANG(STR(?x), "es") AS ?x2)

    service <http://id.sgcb.mcu.es/sparql> {
          ?su a skos:Concept.
  ?su skos:prefLabel ?x2.
    ?su skos:inScheme
<http://id.sgcb.mcu.es/Autoridades/conceptScheme>.
    ?su skos:inScheme
<http://id.sgcb.mcu.es/Autoridades/subjectsNames>
    }
}
```

Tabla 25. SPARQL para enriquecer las materias con las de LEMB

INDICE DE ILUSTRACIONES Y TABLAS

ILUSTRACIONES

TABLAS