FACULTAD DE INGENIERIA

ESCUELA DE INGENIERIA EN SISTEMAS

"Descripción de fuentes de datos heterogéneas utilizando tecnologías semánticas"

Tesis de grado previa a la obtención del título:
Ingeniero en Sistemas

Autores:

Adrián Francisco Angüisaca Landivar
C.I. 0105507503

Juan Pablo Japa Loja
C.I. 0105951271

Director de Tesis:
Ing. Marco Andrés Tello Guerrero
C.I. 0704166818

Co-Director de Tesis
Ing. Víctor Hugo Saquicela Galarza
C.I. 0103599577

Cuenca - Ecuador 2018

Resumen

La Web Semántica, plantea potenciales oportunidades para dotar de significado a los contenidos web. Las ontologías constituyen una de las principales herramientas para especificar explícitamente los conceptos de un dominio concreto, sus propiedades y sus relaciones; de manera que la información se publique en formatos que sean entendibles por agentes máquinas que pueden localizar y gestionar de forma precisa la información.

En esta tesis se presenta una aplicación para la generación un modelo ontológico común, el cual describe diferentes fuentes de datos mediante sus metadatos, específicamente se trabajó con fuentes de tipo Base de datos, CSV, XML y EXCEL. Para ello, se analizó diferentes ontologías de descripción de metadatos, entre las cuales se tienen DCAT, PHDD y DISCO. Estas tres fuentes se unieron en un solo modelo, sobre el cual se realizaron algunas modificaciones, siendo la más importante la incorporación de una estructura, la cual me permita describir los diferentes tipo de datos que tienen los atributos de las fuentes. Después se procedió a la creación de un modelo relacional común, donde se almacena temporalmente los metadatos extraídos, para su posterior mapeo con el modelo ontológico común. Finalmente se procedió a generar un archivo RDF sobre el modelo ontológico común y publicar el mismo para su explotación.

Para validar experimentalmente el modelo creado, se planteó un escenario de integración de varias fuentes de datos, donde se realizo una comparación haciendo consultas al modelo ontológico común y haciendo una inspección de forma manual a los metadatos y datos de dichas fuentes, esto con la finalidad de obtener la utilidad del modelo ontológico común. Concluyendo que mediante el modelo ontológico común el usuario encargado de la integración puede extraer las posibles asignaciones semánticas en términos de integración de datos, de forma fácil y a un menor costo, ya que lo realiza sobre un solo repositorio común.

Palabras clave: ONTOLOGÍA,WEB SEMÁNTICA, RDF,INTEGRACIÓN, SPARQL, ANOTACION SEMANTICA, LINKED DATA, DCAT, PHDD.

Abstract

The Semantic Web, propound potential opportunities to give meaning to web content. Ontologies are one of the main tools to explicitly specify the concepts of a particular domain, its properties and its relationships; so that the information is published in formats that are understandable by machine agents that can locate and manage the information accurately.

This thesis presents an application for the generation of a common ontological model, which describes different data sources through its metadata, specifically worked with sources such as Database, CSV, XML and EXCEL. For this, different ontologies of metadata description were analyzed, among which DCAT, PHDD and DISCO are available. These three sources were united in a single model, on which some modifications were made, the most important being the incorporation of a structure, which allows me to describe the different types of data that have the attributes of the sources. Afterwards, a common relational model was created, where the extracted metadata is temporarily stored, for its subsequent mapping with the common ontological model. Finally, we proceeded to generate an RDF file on the common ontological model and publish the same for its exploitation.

In order to experimentally validate the created model, an integration scenario was set up for several data sources, where a comparison was made by consulting the common ontological model and manually inspecting the metadata and data of said sources, this with the purpose to obtain the utility of the common ontological model. Concluding that through the common ontological model, the user in charge of integration can extract the possible semantic assignments in terms of data integration, easily and at a lower cost, since it is done on a single common repository.

Keywords: ONTOLOGY, SEMANTIC WEB, RDF, SPARQL, SEMANTIC ANNOTATION, LINKED DATA, SELECTION OF ONTOLOGIES.

Índice general

Índice de figuras

Índice de cuadros

Cláusula de licencia y autorización para publicación en el Repositorio Institucional

Adrián Francisco Angüisaca Landivar en calidad de autor y titular de los derechos morales y patrimoniales del trabajo de titulación "Descripción de fuentes de datos heterogéneas utilizando tecnologías semánticas", de conformidad con el Art. 114 del CÓDIGO ORGÁNICO DE LA ECONOMÍA SOCIAL DE LOS CONOCIMIENTOS, CREATIVIDAD E INNOVACIÓN reconozco a favor de la Universidad de Cuenca una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos.

Asimismo, autorizo a la Universidad de Cuenca para que realice la publicación de este trabajo de titulación en el repositorio institucional, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Cuenca, 09 de Mayo del 2018

Adrián Francisco Angüisaca Landivar

C.I: 0105507503

Cláusula de Propiedad Intelectual

Adrián Francisco Angüisaca Landivar, autor del trabajo de titulación "Descripción de fuentes de datos heterogéneas utilizando tecnologías semánticas", certifico que todas las ideas, opiniones y contenidos expuestos en la presente investigación son de exclusiva responsabilidad de su autor.

Cuenca, 09 de Mayo del 2018

Adrián Francisco Angüisaca Landivar

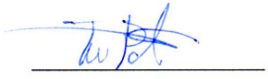
C.I: 0105507503

Cláusula de licencia y autorización para publicación en el Repositorio Institucional

Juan Pablo Japa Loja en calidad de autor y titular de los derechos morales y patrimoniales del trabajo de titulación "Descripción de fuentes de datos heterogéneas utilizando tecnologías semánticas", de conformidad con el Art. 114 del CÓDIGO ORGÁNICO DE LA ECONOMÍA SOCIAL DE LOS CONOCIMIENTOS, CREATIVIDAD E INNOVACIÓN reconozco a favor de la Universidad de Cuenca una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos.

Asimismo, autorizo a la Universidad de Cuenca para que realice la publicación de este trabajo de titulación en el repositorio institucional, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Cuenca, 09 de Mayo del 2018



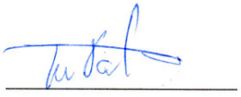
Juan Pablo Japa Loja

C.I: 0105951271

Cláusula de Propiedad Intelectual

Juan Pablo Japa Loja, autor del trabajo de titulación "Descripción de fuentes de datos heterogéneas utilizando tecnologías semánticas", certifico que todas las ideas, opiniones y contenidos expuestos en la presente investigación son de exclusiva responsabilidad de su autor.

Cuenca, 09 de Mayo del 2018



Juan Pablo Japa Loja

C.I: 0105951271

Dedicatoria

A mis padres por su apoyo incondicional, especialmente a mi madre quien con su apoyo y su sabiduría ha sabido guiarme por el buen camino, a mi padre por sus consejos de lucha y valentía. Finalmente a mis hermanos que siempre han estado en las buenas y en las malas y han sabido impulsarme a cumplir mis metas.
Juan Pablo Japa

Dedico esta tesis a mis Padres, Mauro y Clara, pues gracias a su apoyo he logrado este objetivo, la vida no pudo darme unas mejores personas como padres, con su apoyo y amor me han brindado toda la fuerza para llegar al final de mi carrera. Aprovecho este apartado para recordarle cuánto los amo y las infinitas gracias que tengo hacia ellos.
Adrian.

A mis Hermanos Lenin y Karla, que han sabido apoyarme constantemente a lo largo de mi carrera, brindándome un consejo, dándome ánimos o simplemente haciéndome saber que siempre estarán conmigo. Les amo y espero con todo el corazón llegue de igual manera al final de sus carreras. conozco su potencial y se que llegaran a ser los mejores profesionales.
Adrian.

A mi esposa Gladys, que llego a mi vida y se convirtió en mi mundo. Le agradezco por estar a mi lado y por convertirme en una mejor persona, gracias por tantas palabras de aliento y por tanto amor. A mi hija Katherine que me enseñó un amor diferente, el cual es puro, sincero y ha demostrarme que sí existen las personas que no tienen o conocen la maldad. Mi deseo de sacarles adelante me dio el último impulso para culminar este paso tan importante de mi vida. Les amo y de la mano saldremos adelante.
Adrian.

Agradecimientos

Agradecemos a los ingenieros Andres Tello y Victor Squicela por el gran apoyo que nos brindo durante el desarrollo del presente trabajo de tesis. Por dirigir y compartir su conocimiento en este proyecto y ayudarnos a ser unos buenos profesionales. Por último, a agradecemos a todas aquellas personas que de una u otra manera me han apoyado y colaborado para llegar al fin de este proyecto

Capítulo 1

Introducción

1.1. Identificación de un problema

En los últimos años el manejo de la información ha sufrido un cambio drástico, si bien, en años anteriores normalmente se trabajaba sobre un solo sistema de información centralizado, permitiendo a las organizaciones existir y competir en el mercado. La aparición de nuevas tecnologías como el Internet han revolucionado el acceso a los datos digitales a nivel mundial, produciendo un escenario mucho más grande para la obtención de la información, la cual generalmente proviene de fuente de datos heterogéneas. Esto ha obligado que el mundo empresarial actual tenga que trabajar con diferentes, pero coexistentes fuentes de datos o sistemas de información, realizando un gran esfuerzo para a partir de las mismos descubrir oportunidades de negocio y tomar las mejores decisiones. [1].

Al tratar con fuente de datos que poseen modelos de datos diferentes, tanto los costes como los recursos de las organizaciones se incrementan, ya que dichos modelos se deben tratar por separado, utilizando diferentes herramientas y procesos. Así mismo, por cada origen de datos distintos se deben realizar adaptaciones sobre una estructura montada, lo que de igual manera representa cambios muy significativos en la organización. Es en este escenario en donde la integración de fuentes de datos heterogéneas se está volviendo cada vez más indispensable a fin de reducir el esfuerzo en el tratamiento de la información, permitiendo tener una vista unificada de los datos para una accesibilidad idónea, que sirva para las necesidades de negocio. [2].

Si bien el proceso de integración representa una enorme oportunidad para reducir el desarrollo a largo plazo y el costo de mantenimiento de los sistemas heterogéneos, su implementación no es un proceso sencillo debido a diversas razones.

Las fuentes de datos además de ser sintáctica y esquemáticamente heterogéneas, se pueden encontrar en distintas plataformas de hardware lo que aumenta aún más la complejidad de integración. Un segundo conjunto de retos tiene que ver con la forma en que los datos se organizan lógicamente en las fuentes de datos. En su mayor parte las fuentes de datos estructurados se organizan de acuerdo a un esquema. En algunos modelos de datos el esquema especifica un conjunto de tablas cada una con sus respectivos atributos y en otros modelos el esquema se especifica por etiquetas, clases y propiedades etc. Por lo tanto, cuando llegan los datos a partir de múltiples fuentes por lo general se ven muy diferentes aunque representen la misma información [3].

En un proceso de integración, el encargado de la misma, definirá un Esquema Mediador que es el modelo de datos que se quiere lograr con dicho proceso. La relación entre este esquema deseado y los modelos de datos de las diferentes fuentes de datos involucradas, se conocen como asignaciones semánticas y son el componente más importante, pero más complejo y costoso de conseguir dentro del proceso integrador. Al tratar con fuentes de datos totalmente heterogéneas, para poder identificar posibles asignaciones semánticas, se utilizan diferentes herramientas y recursos por cada fuente (denominadas Wrappers), esto con el fin de obtener o extraer información relevante de cada fuente para su análisis y descubrimiento de posibles asignaciones. [4].

Con lo anterior dicho, se puede ver que en un proceso de integración la determinación de asignaciones semánticas requiere de herramientas, recursos y accesos diferentes por cada fuente de datos, ya que las mismas por lo general son esquemáticamente heterogéneas, lo que implica un costo y complejidad muy alto. Debido a esto, en este trabajo se trata de contribuir a la solución de este proceso complejo de integración, mediante procesos automáticos de extracción de metadatos, los cuales serán utilizados en la creación de un modelo ontológico de metadatos común entre las diferentes fuentes de datos. Este modelo de metadatos común, permitirá entre otras cosas estructurar el conocimiento de un modo más formal, aportando significado a los datos. Esto mejora los procesos relacionados a la extracción, recuperación y búsqueda de información, con la finalidad de determinar posibles asignaciones semánticas, ya que se consultará sobre un único modelo de datos, el mismo que almacena la información más relevante de todas las fuentes de datos involucradas, obtenido mediante el análisis previo de sus metadatos y datos.

En la Figura 1.1, se especifica el problema identificado en un proceso de integración y la contribución que se quiere dar con este trabajo, para corregir y optimizar dicho problema.

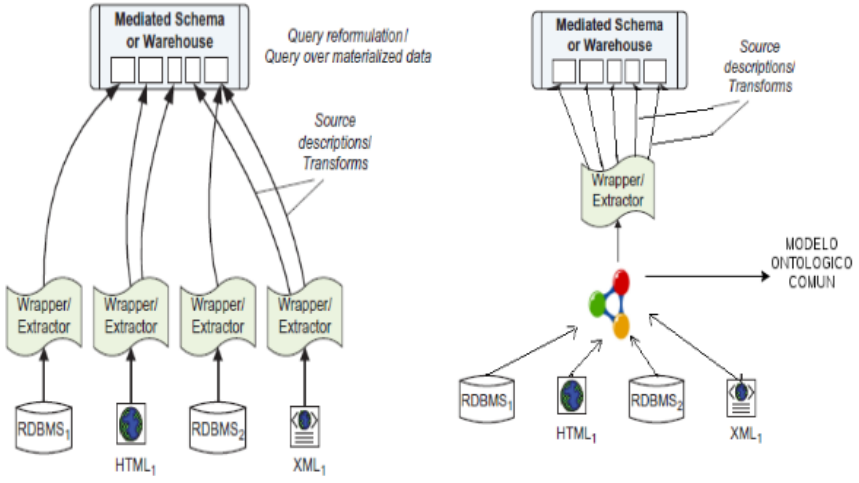


Figura 1.1: Proceso integración normal y optimizado

1.2. Justificación

Con el crecimiento que se está dando últimamente en cuanto a la publicación de información en la gran red de datos enlazados, es importante optar por un modelo ontológico como repositorio común de las diferentes fuentes de datos involucradas en la integración. Porque, al tener publicados los datos, se puede también tener enlaces hacia otros modelos de descripción de fuentes, permitiendo consultar dicha información y a la vez ofreciendo la información de nuestro modelo al exterior.

Sin embargo, la no existencia de un repositorio público dedicado a la descripción de fuentes de datos a través de sus metadatos, causa que en un proceso de integración, la consulta sobre las diferentes fuentes, a fin de determinar asignaciones semánticas, se realice de una manera muy manual. Ya que se debe consultar por separado mediante herramientas y recursos diferentes. Pero incluso, si se llega a obtener de esta manera las asignaciones semánticas, en el proceso de obtención se tuvo que consultar información no relevante, la misma que no aporta mucho significado a la fuente de datos. Lo que resulta una pérdida tanto de recursos como un aumento de costos. Esto resulta mucho más alarmante si se tiene una gran cantidad de fuente de datos. Por ello, se ve la necesidad de la creación y publicación de un modelo ontológico común, donde se publique únicamente la información más relevante de las fuentes de datos involucradas en el proceso de integración y

se proporcione al exterior para su explotación. Esto con la finalidad de obtener de una manera más eficiente y fácil las asignaciones semánticas.

Tomando en cuenta que un repositorio común, que almacene la información más relevante de las diferentes fuentes de datos a integrar, es un área poco explotada en un proceso de integración y mucho menos si se trata de un modelo ontológico. Es importante que este proyecto tome la iniciativa, para que su implementación en el país vaya aumentando y brindando aportes de importancia a la comunidad, especialmente dedicada a la integración de datos.

1.3. Alcance

Se trabajará con diferentes fuentes de datos, específicamente con archivos tipo SQL, CSV, XML y EXCEL. Se consideró archivos de tipo SHAPEFILE sin embargo, los mismos almacenan tanto metadatos como datos en archivos CSV por lo que se consideraron como este tipo de archivos. Para la extracción de metadatos y análisis de datos, se utilizará diferentes herramientas y recursos por cada fuente. Es así por ejemplo para un archivo CSV se utilizaran herramientas como CSVREADER[14] o JAVA.FILE, mientras que para archivos XML herramientas como JDOM[38] y XQUERY[32], más adelante se especifica las diferentes herramientas utilizadas por cada fuente de datos.

Se analizaran diferentes ontologías existentes de descripción de fuentes datos con la finalidad de unirlas, si este es el caso, o modificarlas para obtener un modelo ontológico, que permita describir todas las fuentes antes mencionadas a través de sus metadatos. Para la extracción de los diferentes metadatos de las fuentes, se creará una aplicación JAVA[26] tipo MAVEN[15] en la cual, especificaremos por cada fuente de datos a analizar los parámetros necesarios para la extracción de su información más relevante. Una vez obtenidos los metadatos de cada fuente, es necesario almacenar estos temporalmente antes de su mapeo con el modelo ontológico y posterior generación RDF, por lo cual se creará un modelo relacional común que sirva como almacenamiento temporal de todos los metadatos de las diferentes fuentes. Finalmente se realizará el mapeo entre el modelo relacional temporal y el modelo ontológico común para la generación de RDF y su posterior explotación.

En la figura 1.2, se puede observar un resumen de todo el alcance del presente trabajo.

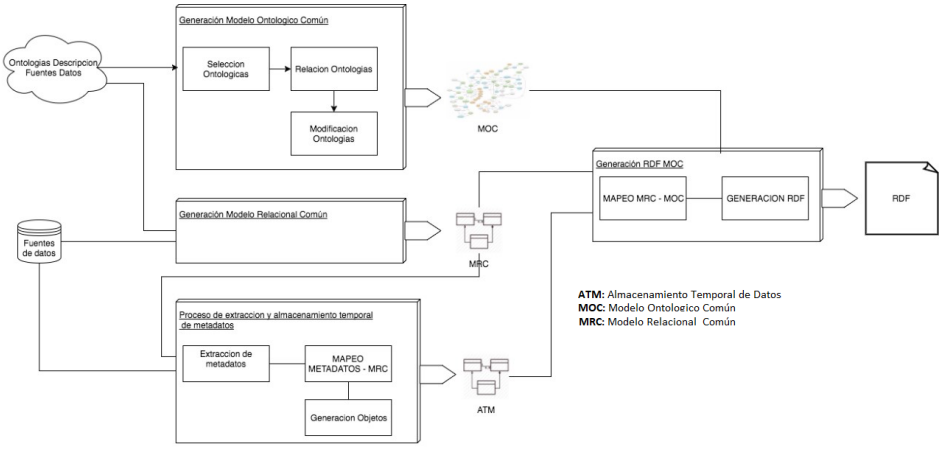


Figura 1.2: Proceso planteado para la generación del modelo ontológico común

1.4. Objetivos Generales

Analizar, diseñar, implementar, publicar y explotar un modelo ontológico común para la descripción de fuentes de datos heterogéneas mediante sus metadatos

1.5. Objetivos Específicos

1. Extraer metadatos de diferentes fuentes de datos heterogéneas.
2. Selección o extensión de un modelo ontológico que permita describir las fuentes de datos mediante sus respectivos metadatos.
3. Crear un modelo relacional general que permita almacenar temporalmente los metadatos extraídos.
4. Realizar mapeo (matching) entre los metadatos obtenidos (modelo objetos general) y el modelo ontológico generado.
5. Generar el RDF correspondiente aplicando el modelo ontológico, el modelo relacional temporal y el matching entre estos.

6. Publicar la ontología generada, en algún repositorio RDF.
7. Obtener posibles asignaciones semánticas mediante la explotación del RDF publicado.

Capítulo 2

Marco Teórico

En el presente capitulo se abarcará los términos claves a los cuales se hará referencia en el presente trabajo. Ademas incluye conceptos básicos sobre las metodologías y herramientas utilizadas.

2.1. Formato de las fuentes de información

A continuación se describe los formatos mas usados para almacenar información, entre ellos esta base de datos, archivos CSV, archivos Excel, archivos XML, entre otros.

2.1.1. Base de Datos

Se trata de una colección de información organizada que permite almacenar, modificar, eliminar y recuperar datos que son almacenados en memoria secundaria, además una base de datos tradicional se organizan por campos, registros y archivos. Es importante destacar que una base de datos tiene diferentes enfoques para lograr almacenar la información depende mucho de la organización donde se llevara acabo la ejecución. En conclusión una base de datos se adapta a las necesidades de la organización, ya sea con soluciones de base de datos de pago o gratuitas.[25]

Los sistemas gestores de bases de datos mas conocidos son:

- Oracle[40]
- Mysql[41]
- Postgres[42]

2.1.2. CSV(Comma Separated Value)

Se trata de un documento de formato abierto, se utiliza para representar datos en forma de tabla. Este tipo documento esta formado por columnas y filas, donde las columnas son separadas por comas o punto y coma, y las filas por saltos de línea.[23]

2.1.3. Archivos EXCEL

Excel es un programa informático desarrollado y distribuido por Microsoft Corp. Este permite realizar tareas contables, financieras gracias a sus funciones. Además es utilizado para almacenar grandes cantidades de información como si fuera una base de datos

Esta aplicación permite leer varios tipos de archivos entre los mas importantes están: xls, xlsx, csv entre otros. Los archivos .xls .xlsx contiene hojas de calculo que permiten realizar operaciones aritméticas. Estos datos están dispuestos en forma de tablas.[24]

2.1.4. Extensible Markup Language (XML)

Un archivo XML esta compuesto por un conjunto de reglas para determinar etiquetas semánticas. Se caracteriza principalmente por ser un lenguaje que se utiliza para decir algo acerca de otro. Los datos de un archivo de tipo XML no depende del hardware o software para su almacenamiento. Por último, el XML es una de esas herramientas que a pesar de su poca complejidad esconden un gran potencial, gracias a que es fácil de usar e innegablemente muy útil. [25]

2.1.5. Shapefile

Un archivo shapefile o ESRI shapefile, es un formato estándar para el intercambio de información entre los sistemas de información geográfica. Este formato hace uso de puntos, líneas y polígonos para representar entidades geográficas.[21]

Está compuesto como mínimo por 3 archivos, donde cada uno de estos puede tener hasta 2GB de tamaño, estos archivos son:

- shp: Almacena las entidades geométricas de los objetos.
- shx: Almacena el índice de las entidades geométricas.
- dbf: Es la base de datos, almacena la información de los atributos de cada elemento en formato dBase14

Es importante destacar que los archivos shapefile deben tener el mismo nombre cada uno con su respectiva extensión, por ejemplo:

- Archivo Principal: cantonesAzuay.shp
- Archivo de Índice: cantonesAzuay.shx
- Tabla de Atributos: cantonesAzuay.dbf

2.2. Herramientas de extracción de metadatos

A continuación se describen algunas herramientas que han sido utilizadas para la extracción de metadatos de las diferentes fuentes de datos mencionadas.

2.2.1. API Geotools

Es una librería de software libre que ofrece gestionar archivos geoespaciales. Por ejemplo una de las funciones es permitir leer un sistema de coordenadas de un archivo shapefile.

Geotools está escrito en el lenguaje de programación Java, desarrollado y mantenido por una comunidad de usuarios. Su arquitectura y diseño modular lo hacen fácilmente extensible para la adición de nuevas funcionalidades, favorecen su utilización como base para el desarrollo de otras aplicaciones.[6]

2.2.2. JDBC (Java Database Connectivity)

Es una librería que contiene un conjunto de interfaces JAVA[26] y métodos que gestionan la conexión a un modelo específico de base de datos. Por lo tanto permite la ejecución de operaciones sobre una base de datos desde el lenguaje de programación JAVA. Vale la pena decir que se debe utilizar el dialecto SQL[27] del modelo de base de datos que se vaya a utilizar.[22]

2.2.3. Apache Tika

El kit de herramientas Apache Tika detecta y extrae metadatos y texto de más de mil tipos de archivos diferentes (como PPT, XLS y PDF). Todos estos tipos de archivos se pueden analizar a través de una sola interfaz, lo que hace que Tika sea útil para la indexación de motores de búsqueda, el análisis de contenido, la traducción y entre otros.[18] A continuación se destaca las interfaces más importantes que contiene esta librería:

- **Interface org.apache.tika.parser.Parser:** Es la parte clave de Apache Tika. Ofrece mecanismos simples para la extracción de metadatos y texto estructurado de todo tipo de documentos.
- **Interface org.apache.tika.detect.Detector:** Permite detectar el tipo de contenido (formato de entrada), además permite detectar el lenguaje del documento.

2.2.4. CSV Reader

Es una librería que permite leer un archivos CSV. Esta librería tiene la característica de poseer analizadores basados en Stream[28] permitiendo un rendimiento máximo y reduciendo los recursos. Permite así leer archivos de cualquier tamaño, incluso permite leer archivos con decenas o cientos de gigabytes.[14]

2.2.5. JDOM

Es una librería de código abierto, ofrece parseo, búsquedas, modificación, generación y serialización de documentos XML de manera sencilla para cualquier aplicación JAVA. Es importante destacar que esta librería está optimizado para el lenguaje de programación JAVA.

JDOM representa un documento XML como un árbol completo disponible todo el tiempo y compuesto por elementos, atributos, comentarios, instrucciones, nodos y secciones. JDOM puede acceder y modificar cualquier parte del árbol, en cualquier momento y los diferentes tipos de nodos del árbol son representados por clases concretas.[38]

2.3. Herramienta para generación RDF

A continuación se describe la herramienta que han sido utilizada para la generación RDF sobre el modelo ontológico común.

2.3.1. Apache Jena

Es una librería destinada a la creación, manipulación de ontologías y la generación de su respectiva salida en RDF, además de permitir expandir los vocabularios existentes de RDF; para lograr esto, Jena ofrece funciones que retornan información de todas las tripletas contenidas en un modelo ontológico, el cual puede ser creado desde cero o cargado desde un archivo independiente. Un modelo ontológico cargado en Jena, contendrá nodos que deberán estar relacionados en una sola dirección.[11]

2.4. Web Semántica

La Web Semántica es una web extendida, dotada de mayor significado en la que cualquier usuario en Internet podrá encontrar respuestas a sus preguntas de forma más rápida y sencilla gracias a una información mejor definida. Al dotar a la web de más significado y, por lo tanto, de la web semántica, se pueden obtener soluciones a problemas habituales en la búsqueda de información gracias a la utilización de una infraestructura común, mediante la cual, es posible compartir, procesar y transferir información de forma sencilla. Esta web extendida y basada en el significado, se apoya en lenguajes universales que resuelven los problemas ocasionados por una web carente de semántica en la que, en ocasiones, el acceso a la información se convierte en una tarea difícil y frustrante.[5]

2.4.1. RDF (Resource Description Framework)

RDF es un modelo estándar para el intercambio de datos en la web. RDF tiene características que facilitan la fusión de datos incluso si los esquemas subyacentes difieren. Este modelo respalda la evolución de los esquemas a lo largo del tiempo sin la necesidad de cambiar todos los datos.

RDF almacena datos como sentencias(tripleta objeto-atributo-valor), donde lo más importante de una sentencia son los recursos, propiedades y valores de las propiedades. La sintaxis de un RDF es generalmente la de un XML.[7]

- **Recurso:** Un recurso se define como cualquier cosa definida por una URI(Uniform Resource Identifier).[29] La URIs permite enlazar los recursos que están en la web a través de sus propiedades.
- **Propiedades:** Básicamente es la relación para un recurso. El principal objetivo de la propiedad es desarrollar un vocabulario común y un esquema de relaciones similares para diferentes dominios.
- **Valores de Propiedades:** Se define como el valor que tiene dicha propiedad, por ejemplo puede ser un valor literal de tipo String o recursos web de tipo URI.

La figura 2.1 presenta una sentencia en forma de un grafo, donde esta compuesta por dos nodos(sujeto y objeto) unidos por un arco(predicado), donde los arcos representan propiedades y los nodos recursos.

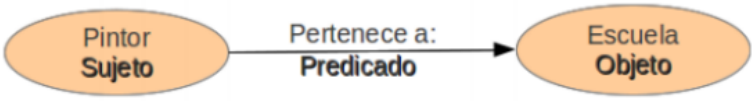


Figura 2.1: Sentencia RDF en Grafo

2.4.2. Protocol and RDF Query Language(SPARQL)

SPARQL es un lenguaje de consultas de la W3C(World Wide Web Consortium)[30] para RDF. Esto indica que es un protocolo para extraer datos desde un grafo RDF.[12]

En la Figura 2.2 presenta la sintaxis donde la palabra SELECT sirve para definir los datos que deben ser devueltos. La palabra FROM indica los datos sobre cuales se ejecutará la consulta y la palabra WHERE indica el patrón sobre el que se filtraran las tripletas RDF.

2.5. Ontologías para la descripción de fuentes de datos

A continuación se describe algunos modelos ontológicos, que han sido utilizados para generar el modelo ontológico común de descripción de fuentes de datos.

2.5.1. DCAT(Data Catalog Vocabulary)

DCAT es un vocabulario RDF diseñado para facilitar la interoperabilidad entre los catálogos de datos publicados en la Web. Al usar DCAT para describir conjutos de datos en catalogos de datos, los editores aumentan la visibilidad y permiten que las aplicaciones consuman fácilmente los metadatos de múltiples catálogos. Además, permite la publicación descentralizada de catálogos y facilita la búsqueda de conjuntos de datos federados en todos los sitios.[34]

DCAT define tres clases principales(tabla 2.1)

2.5.2. PHDD(Physical Data Description)

PHDD es un vocabulario RDF para la descripción de las propiedades físicas de los datos existentes o publicados en formato rectangular(Tablas). Incluidos valores

Sintaxis básica de una consulta SPARQL	
Prologue (optional)	BASE <iri> PREFIX prefix: <iri> (repeatable)
Query Result forms (required, pick 1)	SELECT (DISTINCT) sequence of ?variable SELECT (DISTINCT)* DESCRIBE sequence of ?variable DESCRIBE * CONSTRUCT { graph pattern } ASK
Query Dataset Sources (optional)	Add FROM FROM NAMED
Graph Pattern (optional, required for ASK)	WHERE { tripleta}
Query Results Ordering (optional)	ORDER BY ...
Query Results Selection (optional)	LIMIT n, OFFSET m

Figura 2.2: presenta la Sintaxis de una consulta SPARQL

Propiedad	**Descripción**
dcat:Catalog	Representa el catálogo
dcat:Dataset	Representa un conjunto de datos en un catálogo
dcat:Distribution	Representa una forma accesible de un conjunto de datos como, por ejemplo, un archivo descargable, una fuente RSS, una archivo rectangular o un servicio web que proporciona los datos.

Cuadro 2.1: Definición de DCAT

separados por comas (CSV) o similares, se centra exclusivamente en las propiedades físicas de los archivos.

PHDD podría usarse de manera independiente (es decir, agregando información relevante a archivos de formato retangular) o en una descripción más completa de datos junto con las posibilidades de DDI-RDF Discovery (DISCO)[31] y Data Catalog Vocabulary (DCAT), lo que permite la creación de repositorios de datos que proporcionan metadatos para la descripción de colecciones asi como el descubrimiento y procesamiento de los datos.[10]

Las propiedades principales del vocabulario se puede observar en la tabla 2.2

Propiedad	**Descripción**
phdd:Table	Una tabla, que podría ser un archivo rectangular con valores separados por caracteres (CSV) o un archivo rectangular con longitud de registro fija. Esta puede ser una subclase de dcat:Distribution en el Catálogo de Datos de Vocabulario (DCAT)[34]
phdd:TableDescription	Propiedades de la tabla descritas por parámetros significativos.
phdd:TableStructure	Propiedades de la tabla descritas por los parámetros predeterminados de los valores de los datos.
phdd:Column	Una columna de la tabla. Otros términos son variable o atributo.
phdd:ColumnDescription	Descripción detallada de una columna.

Cuadro 2.2: Definición de PHDD

2.5.3. DDI-RDF Discovery Vocabulary (Disco)

Disco define un vocabulario de esquema RDF que permite el descubrimiento de datos de investigación y encuestas en la Web. Se basa en los formatos DDI XML de DDI Codebook y DDI Lifecycle.[35]

Esta especificación está diseñada para soportar el descubrimiento de conjuntos de microdatos y metadatos relacionados, utilizando tecnologías RDF en la Web de Datos Vinculados. Permite la identificación programática, de los conjuntos de datos relevantes para un propósito de investigación específico. [33]

Cuando se trata de entender el contenido del conjunto de datos, esto se hace usando la clase Variable. Las variables (Variable) proporcionan una definición de la columna en un archivo de datos rectangular. Las variables están relacionadas con una representación de alguna forma, que puede ser cualquier tipo de datos normales (fecha y hora, numéricos, textuales, etc.).

Las principales propiedades del vocabulario, que serán de utilidad en el desarrollo del modelo ontológico común de metadatos, se pueden observar en la tabla 2.3

Propiedad	Descripción
disco:Variable	Proporcionan una definición de la columna en un archivo de datos rectangular.
disco: RepresentedVariable	Abarca partes de variables que pueden ser reutilizable por otras variables. Es decir un variable puede estar basada en otras variables.

Cuadro 2.3: Definición de DISCO

2.6. Definición Proceso de integración de Datos

La integración de datos la podemos definir como el proceso de combinar datos que residen en diferentes fuentes y permitirle al usuario final tener una vista unificada de todos los datos[3].

2.6.1. Arquitectura de un sistema de integración de datos

Existe una variedad de arquitecturas posibles para la integración de datos, pero en términos generales, la mayoría de los sistemas se ubican en algún lugar entre el almacenamiento e integración virtual. En el almacenamiento, los datos de las fuentes de datos individuales, se cargan y materializan en una base de datos física (llamada almacén), donde se pueden responder las consultas sobre los datos. En la integración virtual, los datos permanecen en las fuentes y se accede a ellos según sea necesario en el momento de la consulta. A pesar de las diferencias en el enfoque, muchos de los desafíos difíciles se comparten a través de estas arquitecturas.

Componentes del sistema de integración de datos

A continuación se describe los componentes utilizados en el enfoque virtual de integración. En la parte inferior de la figura 2.3, se muestra las **fuentes de datos**. Las fuentes de datos pueden variar en muchas dimensiones, tanto en el modelo de datos que las sustenta como en los tipos de consultas que requieren para obtener su información. Los ejemplos de fuentes estructuradas incluyen por

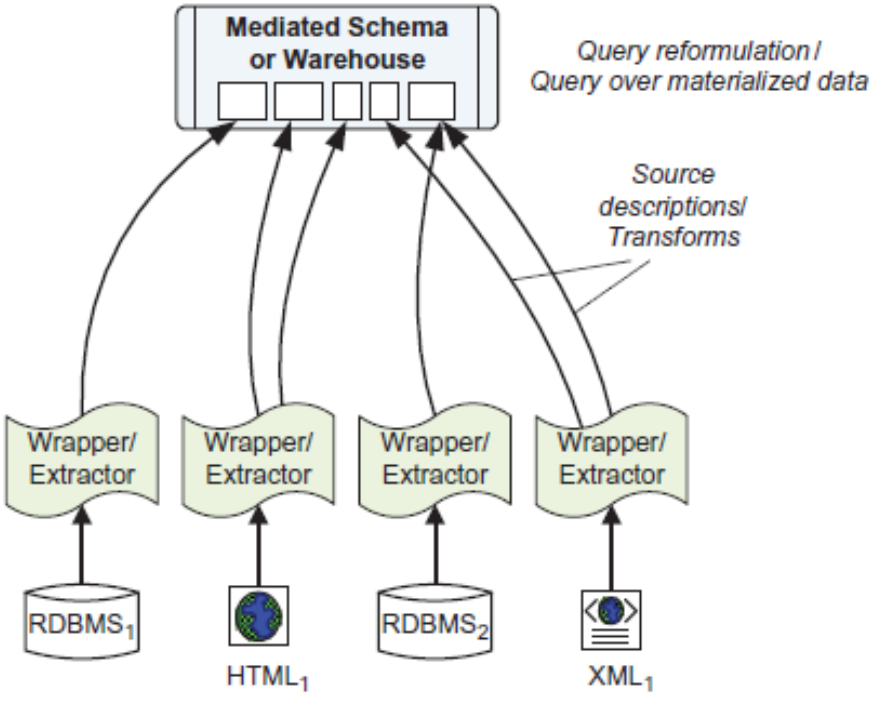


Figura 2.3: Componentes lógicos de sistemas de integración de datos

ejemplo sistemas de bases de datos con capacidades SQL o bases de datos XML con una interfaz *XQuery.*[32]

Por encima de las fuentes de datos se encuentran los programas cuya función es comunicarse con estas fuentes de datos. En la integración de datos virtuales, estos programas se denominan ***wrappers***, y su función es enviar consultas a una fuente de datos, recibir respuestas y posiblemente aplicar algunas transformaciones básicas en la respuesta.

El usuario interactúa con el sistema de integración de datos a través de un único esquema, llamado esquema mediador. El esquema mediador(Mediated Schema) es construido para la aplicación de integración de datos y contiene sólo los aspectos del dominio que son relevantes para la aplicación. Como tal, no contiene necesariamente todos los atributos que vemos en las fuentes, sino sólo un "subconjunto"de

ellos. En el enfoque virtual, el esquema mediador "no está destinado a almacenar ningún dato". Es simplemente un esquema lógico que se utiliza para plantear consultas de los usuarios (o aplicaciones) que emplean el sistema de integración de datos.

La clave para construir una aplicación de integración de datos son los ***Source Descriptions***, se encargan de conectar el esquema mediador y los esquemas de las fuentes. El componente principal de ***Source Descriptions*** son las asignaciones semánticas, que especifican cómo los atributos en las fuentes corresponden a los atributos en el esquema mediador (cuando tales correspondencias existen). Es importante enfatizar que solo se requiere la especificación de asignaciones entre las fuentes de datos y el esquema mediador y no entre cada par de fuentes de datos.

2.7. Técnicas de Schema Mapping

Formalmente, un mapeo de esquema es un conjunto de expresiones que describen una relación entre un conjunto de esquemas (comúnmente dos). En el caso del proceso de integración , las asignaciones de esquema describen una relación entre el esquema mediador y el esquema de las fuentes.

Cuando una consulta es formulada en términos del esquema mediador, usamos los mapeos para reconstruir la consulta de tal manera que sea apropiada sobre las fuentes.

A continuación se exponen los enfoques más utilizados en schema mapping. Figura 2.4

2.7.1. Global-as-View

Global-as-View (GAV), adopta un enfoque muy intuitivo para especificar mapeos de esquema. GAV define el esquema mediador como un conjunto de vistas sobre las fuentes de datos. El esquema mediador a menudo se conoce como esquema global. [36]

GAV se recomienda usar cuando se plantea la integración en entornos donde las fuentes de información no cambian regularmente.

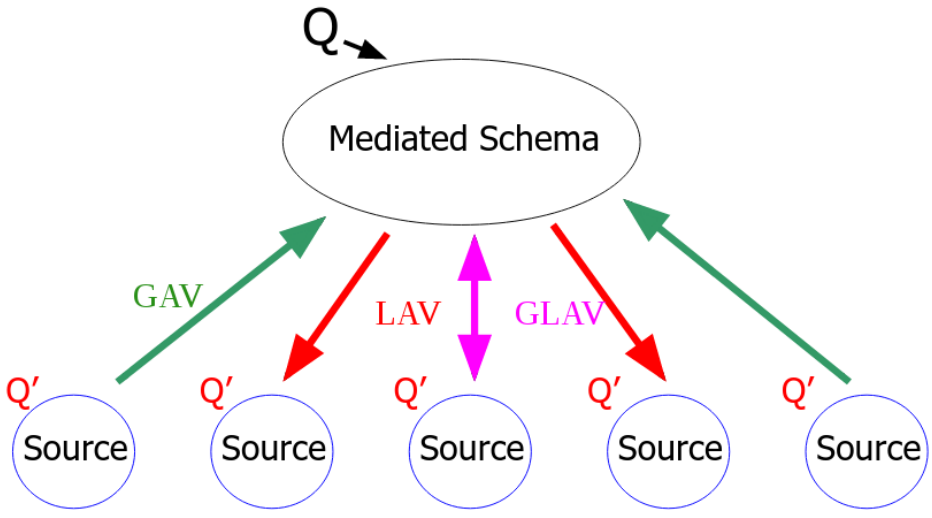


Figura 2.4: Principales tecnicas de Schema Mapping

2.7.2. Local-as-View

Local-as-View (LAV) toma el enfoque opuesto a GAV. En lugar de especificar cómo calcular tuplas del esquema mediador, LAV se centra en describir cada fuente de datos a la posible con precisión y de forma independiente de cualesquiera otras fuentes. [36]

LAV se recomienda usar cuando se plantea la integración en entornos donde las fuentes de información son propensas al cambio.

2.7.3. LAV vs. GAV

GAV tiene como ventaja la ejecución de las consultas globales de forma eficiente, mientras que LAV permite agregar o eliminar fuentes de información de forma simple.

Parametro	GAV	LAV
calidad	Depende de que también se haya compilado las fuentes en el esquema global	depende de la caracterización de las fuentes
Extensibilidad	Implica rehacer el esquema global	Muy simple, basta con agregar las fuentes de datos
Procesamiento de consultas	Es más sencillo pues la información modelo es la suma de todas las vistas de todos los orígenes de información	Necesita más procesamiento pues se debe replantear la consulta traducida a los términos de los orígenes de información

Cuadro 2.4: Comparación entre GAV y LAV

2.8. Publicación

El RDF generado debe ser publicado para su posterior explotación por medio de consultas. Por lo tanto debe ser cargado o subido en un servidor de tripletas, el cual gestionará la información de los archivos RDF. A continuación se enumeran algunos de los servidores de tripletas.

- Virtuoso
- Apache Marmotta
- Fuseki

2.8.1. Apache Mamotta

Apache marmotta es un servidor de linked data(LD), servidor de SPARQL y un entorno de desarrollo de linked data. La arquitectura modular del servidor hace posible la implementación únicamente del módulo que se requiera. Marmotta ofrece una colección de bibliotecas para el acceso a recursos linked data y consultas linked data.[17]

Modulos SPARQL

Ofrece una interfaz unificada para consultas y actualización basada en Squebi y visualización de datos basada en Sgvizler.[39]

Contiene los siguientes componentes:

- Kiwi, es un triplestore construido encima de una base de datos relacional.

- LDPATH, un lenguaje de ruta para navegar a través de los recursos de Linked Data.
- LDClient, un cliente Linked Data que permite la recuperación de los recursos remotos a través de diferentes protocolos.
- LDCache, un sistema de caché que recupera automáticamente los recursos mediante el uso de LDClient internamente.

Capítulo 3

Descripción del modelo ontológico común para descripción de fuente de datos.

3.1. Selección de ontologías

En esta sección, se menciona diferentes ontologías que permiten describir de forma adecuada las fuentes de datos existentes a través de sus metadatos. Algunos de lo modelos ontológicos que se analizó son:

- ***Vocabulary of Interlinked Datasets (void)*:** Es un vocabulario RDF para expresar metadatos RDF sobre conjuntos de datos. Pretende ser un puente entre los editores y usuarios de datos RDF. Además permite el descubrimiento y el uso de conjuntos de datos vinculados. Un conjunto de datos vinculada es una colección de datos, publicados y mantenidos por un único proveedor, disponible como RDF en la Web, donde al menos algunos de los recursos en el conjunto de datos se identifican mediante URI(Uniform Resource Identifier).[33]
- ***Data Catalog Vocabulary (DCAT)*:** Es un vocabulario RDF diseñado para facilitar la interoperabilidad entre los catálogos de datos publicados en la Web, mediante su uso se puede describir conjuntos de datos en los catálogos de datos. Además, permite la publicación de catálogos de forma descentralizada y facilita la búsqueda federada de datos a través de sitios. DCAT no hace ninguna suposición sobre el formato de los datos que se

describen en el catálogo. Otros vocabularios, complementarios pueden ser utilizados junto con DCAT para proporcionar información más detallada específica del formato.[34]

- ***Physical Data Description (PHDD)*:** Proporciona una descripción física de datos existentes o publicados en un formato rectangular (tablas). Los datos podrían ser representados en los registros con valores de caracteres separados (CSV), en registros de longitud fija y en otros archivos similares. PHDD podría ser utilizado independiente o junto con vocabularios relacionados como el catálogo de datos de vocabulario (DCAT) o DDI - RDF Descubrimiento (Disco).[10]
- ***Metadata Authority Description Schema (MADS)*:** Los MADS / RDF está diseñado como un modelo de datos de la autoridad y de vocabulario de datos utilizados dentro de la comunidad de la biblioteconomía y documentación (LIS), que incluye a museos, archivos y otras instituciones culturales. Por ejemplo, MADS / RDF proporciona un medio para registrar los datos desde el formato de lectura mecánica Catalogación (MARC) Autoridades en RDF para su uso en aplicaciones semánticas y proyectos de *Linked Data*.
- ***Dataset Catalog Vocabulary (DS)*:** Este vocabulario se utiliza para el modelado de los catálogos de conjuntos de datos y sus relaciones con los conjuntos de datos. Similar al vocabulario DCAT.[34]
- ***DDI-RDF Discovery Vocabulary (DISCO)*:** Disco define un vocabulario de esquema RDF que permite el descubrimiento de datos de investigación y encuestas en la Web. DISCO está diseñada para admitir el descubrimiento de conjuntos de microdatos y metadatos relacionados utilizando tecnologías RDF en la Web de Datos Vinculados. [31]

Una vez analizados los distintos modelos ontológicos, se llegó a la conclusión que individualmente, ninguno permite describir adecuadamente las diferentes fuentes de datos mediante sus metadatos, tanto de una forma lógica como física. Razón por la cual se optó por trabajar con varios modelos, los cuales permitan crear un poderoso repositorio de metadatos. Específicamente se optó por los modelos DCAT, PHDD y DISCO. Se eligieron estos tres modelos, principalmente por la relación directa que guardan entre si. Dicha relación se explicará más adelante.

Se decidió trabajar con el modelo DCAT debido a que permite describir catálogos de datos, permitiendo asociar diferentes fuentes de datos a un catálogo común para su publicación, si este es el caso. Mediante DCAT se permiten que las aplicaciones fácilmente consuman metadatos de múltiples catálogos. Además, permite la publicación de catálogos de forma descentralizada y facilita la búsqueda de datos de manera federada. Adicionalmente este modelo proporciona la facilidad de

trabajar con otros vocabularios entre los cuales están PHDD y DISCO.

Si bien el vocabulario DCAT permite describir de manera general diversas fuentes de datos especificando autor, fecha de creación, *keywords*, entre otros; y asociar las mismas a un catálogo común. Es necesario describir además físicamente el contenido (datos) y la estructura del *dataset*. El modelo PHDD permite describir las propiedades físicas de un archivo de datos haciendo énfasis en los tipos de formato más comunes: Formatos Rectangular (Tablas), CSV y archivos de longitud de registro fija.

Por su parte DISCO, define un vocabulario RDF que permite el descubrimiento de los datos de investigación y estudio en la Web. El vocabulario aprovecha la especificación DDI [31] para crear una versión simplificada de este modelo para el descubrimiento de los archivos de datos. Además es compatible con la identificación de los conjuntos de datos relevantes para un propósito específico de investigación.

3.1.1. Relación de los modelos seleccionados

Como se menciona anteriormente el uso combinado de los modelos DCAT, PHDD y Disco permite la creación de repositorios de datos que proporcionan los metadatos para la descripción de las colecciones, asi como, para el descubrimiento y procesamiento de los datos. Ademas permite describir varias fuentes de datos tanto de forma lógica como física y asociar las mismas a un catálogo común para su publicación y explotación.

A continuación se describe detalladamente cada modelo ontológico seleccionado, para posteriormente explicar la forma en la que se relacionan.

1. **DCAT (Data Catalog Vocabulary)**

   DCAT (Figura 3.1), es un vocabulario RDF adaptado para representación de catálogos de datos, entre los cuales están catálogos del gobierno, como Data.gov y data.gov.uk. DCAT define tres clases principales:

   - **dcat:Catalog:** representa un Catalogo
   - **dcat:Dataset:** representa un dataset en un Catalogo.
   - **dcat:Distribution:** representa una forma accesible a un conjunto de datos(dataset), por ejemplo datos en formato rectangular (Tablas). Esta clase tiene una relación directa con el vocabulario PHDD)

   El conjunto de datos en DCAT se define como una colección de datos, publicados por un solo agente y disponible para el acceso o descarga en uno

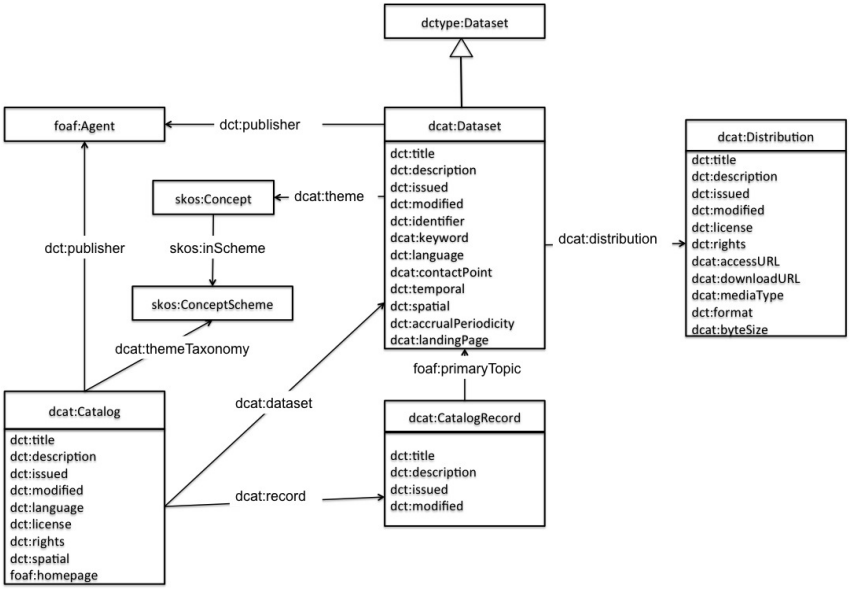


Figura 3.1: DCAT (Data Catalog Vocabulary)

o más formatos. Un conjunto de datos no tiene que estar disponible solo como un archivo descargable. Por ejemplo, un conjunto de datos puede ser accedido mediante archivos en formato rectangular (tablas), la cual viene a ser la relación entre el vocabulario DCAT y PHDD (Figura 3.2)

Es importante destacar la clase dcat:CatalogRecord que describe un conjunto de datos de entrada en el catálogo. Si bien dcat:dataset representa el conjunto de datos en sí , dcat:CatalogRecord representa el registro que describe un conjunto de datos en el catálogo . El uso del CatalogRecord se considera opcional. Se utiliza para capturar información sobre la procedencia del conjunto de datos de entrada en un catálogo.

2. **Physical Data Description (PHDD)**

   PHDD(Figura 3.3), este modelo ontológico provee una descripción de las propiedades físicas de un archivo de datos, enfocándose en los tipos de formato más comunes como: Formato Rectangular (Tablas), CSV y Fixed-record length.

   La estructura básica del vocabulario PHDD lo componen principalmente las siguientes clases:

   - **Table:** Archivo de datos rectangular

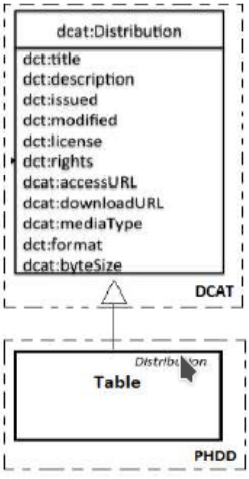


Figura 3.2: DCAT (Relación entre el vocabulario DCAT y PHDD)

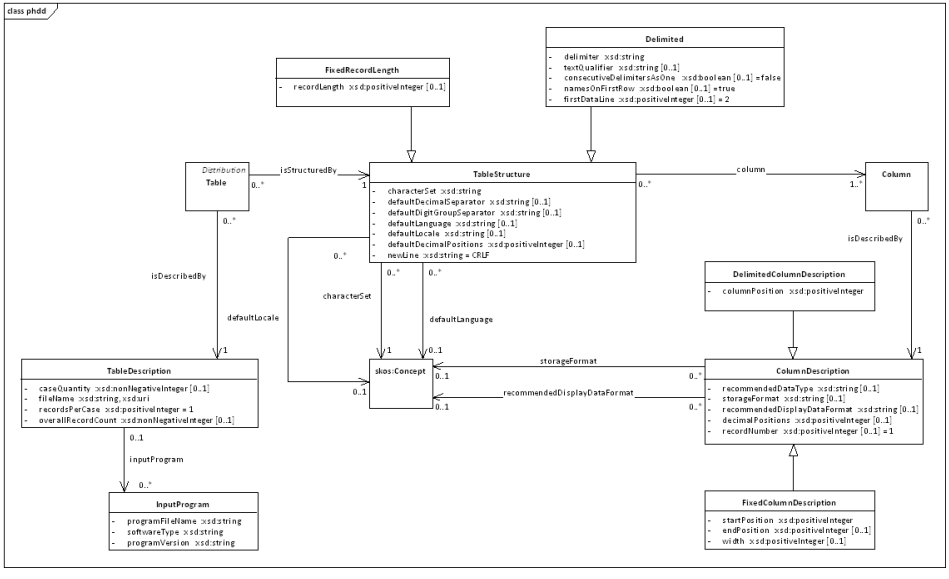


Figura 3.3: Physical Data Description (PHDD)

- **TableStructure:** Representa las propiedades del archivo de formato rectangular, basado en las propiedades predeterminadas de los valores

de los datos. Describe propiedades como: conjunto de caracteres, separador decimal por default que utiliza la tabla, etc. Las clases Delimited así como FixedRecordLength son especializaciones de TableStructure tanto para archivos CSV como archivos FixedRecordLength respectivamente, en las mismas se especifican atributos específicos como 'delimitador' para el archivo CSV así como 'recordLength' para el archivo de longitud fija. En este punto es importante mencionar que para la descripción de las diferentes fuentes de datos anteriormente mencionadas como XML y EXCEL, se deberá modificar el vocabulario extendiendo de TableStructure diferentes clases, en las cuales se especificaran las propiedades especificas de estos tipos de archivos o simplemente para hacer la distinción entre las diferentes fuentes.

- **Column:** Representa las columnas del archivo rectangular (Tabla), esta clase se relaciona directamente con el vocabulario DISCO permitiendo relacionar ambos modelos. (Figura 3.4).

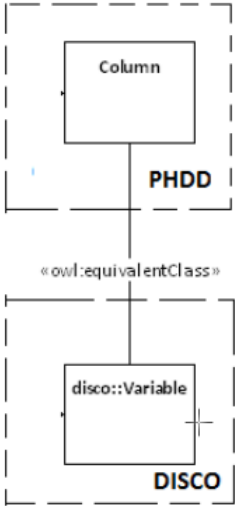


Figura 3.4: Relación PHDD y DISCO

- **ColumnDescription:** Representa las propiedades de cada columna, de esta clase se especializan las clases DelimitedColumnDescription y FixedColumnDescription en las cuales se especifica las propiedades tanto para archivos CSV y FixedRecordLength respectivamente.

De igual manera que en la clase TableStructure, para poder describir los diferentes fuentes de datos se deberá modificar el modelo creando nuevas es-

pecializaciones de la clase ColumnDescription, que servirán para especificar las propiedades específicas de estas fuentes de datos (CSV, EXCEL, etc.). Más adelante se explicara las modificaciones realizadas sobre el modelo.

3. **DDI-RDF Discovery Vocabulary**

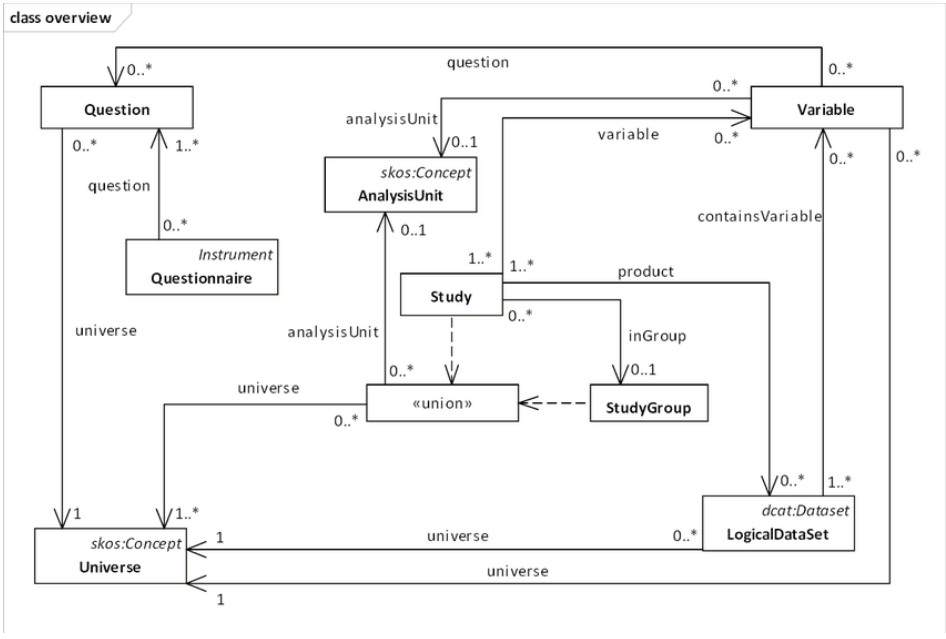


Figura 3.5: DDI-RDF Discovery Vocabulary

DISCO (Figura 3.5) define un vocabulario RDF que permite el descubrimiento de datos de investigación y estudio en la Web.

La estructura del vocabulario esta principalmente formado por las siguientes clases:

- Study: Representa el proceso por el cual un conjunto de datos se generó o se recogió.
- StudyGroup: En algunos casos, cuando la recopilación de datos es cíclico o está en curso, los conjuntos de datos se pueden transformar en un Studygroup, donde cada ciclo de la actividad de recolección de datos produce uno o más conjuntos de datos.

- LogicalDataSet representa el contenido del archivo (conjunto de variables (variable)).
- Variable (Variables) proporcionan una definición de la columna en un archivo de datos rectangular, y pueden asociarlo con un concepto en particular. Las variables están relacionadas con una representación de alguna forma, que puede ser cualquier tipo de datos normales (fecha y hora, numéricos, textuales, etc.).
  La clase Variable permite relacionar directamente el modelo DISCO con el modelo PHDD como se especifica en la Figura 3.4

Como se puede ver el vocabulario DISCO al igual que los modelos anteriormente mencionados permiten describir tanto de forma lógica como física un conjunto de datos relacionándolos hacia una área de estudio e investigación. Sin embargo, para el propósito de este trabajo solo se utilizará una parte del modelo ya que no se pretende relacionar las diferentes fuentes de datos hacia una área especifica de investigación.

En este trabajo se utilizará únicamente una parte específica del modelo DISCO, la misma que me permitirá definir y especificar las columnas de cada archivo de formato rectangular (Tabla), en la figura 3.6 Se muestra la parte del modelo DISCO con el que se trabajara.

En definitiva los modelos DCAT, PHDD y DISCO se pueden relacionar directamente mediante las siguientes clases:

- **dcat:Distribution – phdd:Table** La clase Table del vocabulario PHDD es una especialización de la clase Distribution del vocabulario PHDD.
- **phdd:Column – disco:Variable** La clase Column del vocabulario PHDD se una clase equivalente con la clase Variable del vocabulario Disco.

En la figura 3.7 se muestra la relación básica entre los diferentes modelos, mientras que en el Anexo A se muestra el modelo general completo, resultado de la unión de 3 vocabularios.

3.2. Modificación sobre el modelo ontológico general

Sobre el modelo ontológico general, formado por 3 vocabularios antes mencionados, se deben tomar las siguientes consideraciones:

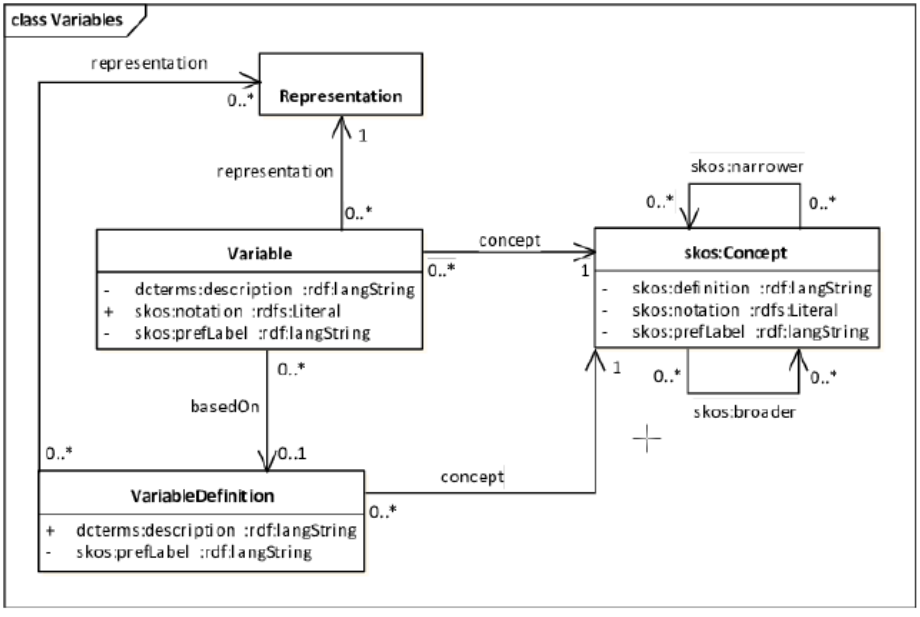


Figura 3.6: DISCO

- En el vocabulario DCAT, no se considera el tipo de formato de la fuente de datos a describir, es decir DCAT no hace ninguna suposición sobre el formato de los conjuntos de datos que se describe en el catálogo
- El vocabulario PHDD está enfocado hacia archivos de formato rectangular(Tablas), archivos CSV y archivos de longitud fija (Fixed Record Length).
- Los diferentes vocabularios analizados y seleccionados están enfocados para su utilización manual, es decir los metadatos de las diferentes fuentes de datos son obtenidos manualmente. En este trabajo la mayoría de los metadatos se obtendrán de manera automática mediante diferentes procesos por lo que la fiabilidad o veracidad de cada metadato obtenido es menor que en una obtención manual.
  Dicho esto se deberán hacer modificaciones sobre el modelo de tal manera que brinde la mayor confiabilidad sobre los metadatos obtenidos.

Con estas consideraciones se realizaron las siguientes modificaciones sobre el modelo:

- Extensión para archivos EXCEL y XML.
  Como se mencionó el modelo PHDD esta orientado hacia archivos de formato rectangular, CSV y Fixed Record Length, razón por la cual se debe

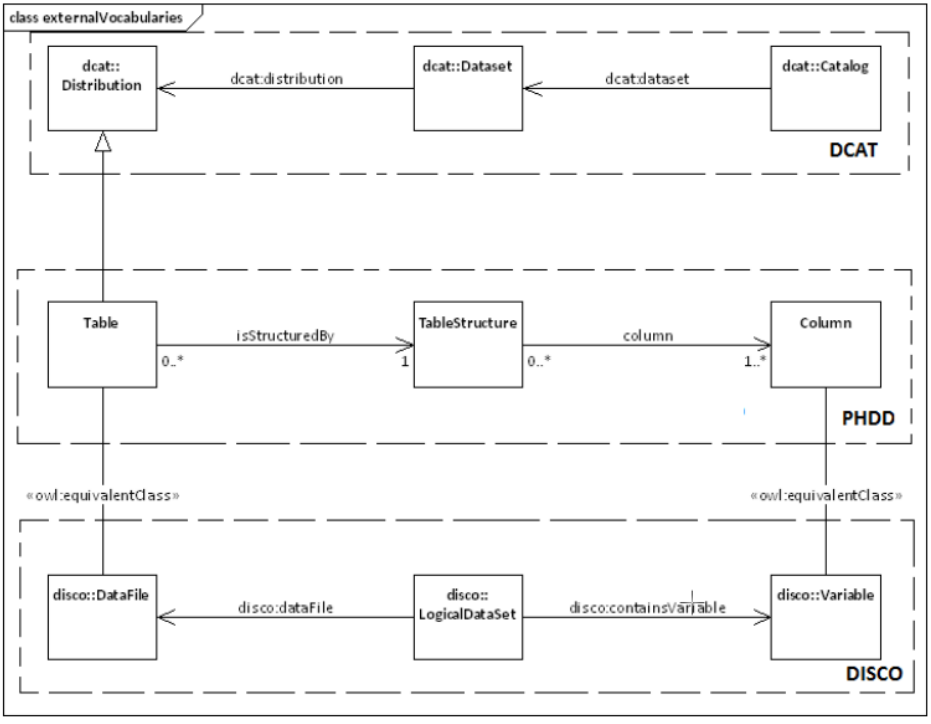


Figura 3.7: Relación básica entre DCAT, PHDD y DISCO

modificar el modelo para poder describir otros tipos de fuentes de datos. Para esto se decidió extender tanto de la clase TableStructure como de la clase ColumnDescription, clases propias para los archivos mencionados. Si bien todos estos nuevos archivos se podrían ser considerados como archivos de formato rectangular es importante la especialización para hacer una diferenciación entre las diferentes fuentes de datos.
De la clase TableStructure se extendió las diferentes clases:

- XMLStrucutre clase en la cual se especificarán los propiedades especificas de un archivo XML.
- EXCELStructure clase en la cual se especificarán los propiedades especificas de un archivo EXCEL.

De la clase ColumnDescription se extendió las diferentes clases:

- XMLColumnDescription clase en la cual se especificarán los propiedades especificas de una columna de un archivo XML.

  - EXCELColumDescription clase en la cual se especificarán los propiedades especificas de una columna de un archivo EXCEL.

- En este punto es importante recalcar que el modelo generado es completamente extensible a futuro, para otros tipos de fuentes datos como: GSON, WebServices, entre otros.

  Otra consideración importante que se debe tomar, es el tipo de dato de cada columna de las diferentes fuentes de datos. Sabiendo que se va a extraer los metadatos de forma automática, el tipo de formato de cada columna que se determine no es 100 % confiable en algunos tipos de archivos. Los tipos de datos en columnas correspondientes a una tabla de base de datos si resulta 100 % confiable ya que esta información se obtendrá directamente del schema de la base de datos mediante el diccionario de datos, sin embargo en archivos como CSV, XML y EXCEL no se tiene la menor certeza de que tipo de dato es cada columna, razón por la cual se debe analizar los datos de cada columna para poder identificarlo.

3.3. Proceso de identificación de tipo de dato de una columna

Para la obtención del tipo de dato de cada columna de una tabla de base datos se consultará directamente el diccionario de datos de la base de datos, obteniendo el formato de columna con una veracidad del 100 %.

En el caso de los archivos CSV, XML y EXCEL se deben analizar los datos de cada columna para poder obtener el tipo de dato.

Para este proceso, se analizará un porcentaje aleatorio de registros de la fuente de datos, por ejemplo de un archivo CSV de 100 registros se podría analizar un 40, 50 o 70 % de registros, de los cuales se obtendrán los diferentes tipos de datos en los que se pueden convertir cada dato de la columna. Por ejemplo, se obtendrá la siguiente información: 70 % de los datos se pueden considerar tipo DATE, el 30 % se puede considerar como String. Proporcionando información que indique que de todos los datos pertenecientes a esa columna, existirá un 30 % que quizá presente algún ruido por lo que no pueden ser considerados como DATE pero si como String.

Toda esta información se proporciona al usuario, puesto que el debe tener la certeza de como tratar los datos, ya que si solo se proporciona la información de que la mayoría de datos son tipo DATE podrían ocurrir o traer problemas futuros ese 30 % de datos de diferente tipo.

Adicionalmente se debe considerar aspectos propios de cada tipo de dato es decir:

- Si la columna es de tipo Entero se debería especificar cual es el valor máximo y el valor mínimo.
- Si la columna es de tipo Date se debería especificar el formato del mismo por ejemplo: dd-MMM-yyy o dd-MM yyyy-HH:MM:SS.
- Si la columna es de tipo Decimal se debería de igual manera especificar el máximo y el mínimo así como otros atributos, por ejemplo el numero de decimales, etc.

En la tabla 3.1, se especifican todos los atributos que se definirán por cada tipo de dato.

Tipo de Dato	**Atributo**	**Descripción**
String	minLength	Longitud de la cadena con menor longitud
	maxLength	Longitud de la cadena con mayor longitud
Integer	numMax	Entero con el valor mas alto.
	numMin	Entero con el valor mas bajo.
	average	Promedio de todos los datos.
Decimal	numMax	Decimal con el valor mas alto.
	numMin	Decimal con el valor mas bajo.
	average	Promedio de todos los datos.
	numMaxDecimal	Mayor cantidad de decimales.
	numMinDecimal	Menor cantidad de decimales.
Boolean	numTrueValue	Numero de datos con valor true
	numFalseValue	Numero de datos con valor de false
Date	formato	Formato de fecha.

Cuadro 3.1: Atributos por cada tipo de dato.

Estas modificaciones sobre el tipo de dato de cada columna brindara al encargado de la integración mayor información de como realizar el tratamiento de los datos. Con todas estas consideraciones se modificó el modelo ontológico, resultado en el modelo ontológico final sobre el que se trabajara Anexo B.

Capítulo 4

Generación del modelo relacional comùn.

En el presente capítulo se definirá el modelo relacional común, el mismo que tiene como objetivo almacenar los diferentes metadatos de las distintas fuentes. Este modelo es el encargado de almacenar temporalmente los metadatos extraídos para su tratamiento previo al mapeo con el modelo ontológico común y posterior generación del RDF.

Los pasos considerados para la generación del modelo relacional común son:

- **Definición del modelo relacional por cada fuente de datos:** En esta sección se explicará brevemente el modelo relacional que se creará o usará por cada fuentes de datos. Estos modelos individuales se usarán posteriormente para la definición de un modelo relacional general.

- **Definición del modelo relacional común:** En esta sección se definirá un modelo relacional general común para las diferentes fuentes de datos, basándose en los modelos individuales de cada fuente.

- **Definición de la estructura StorageFormat:** Como se explicó en el capítulo tres. Una modificación importante sobre las ontologias seleccionadas, es la incorporación de la estructura que almacenará toda la información referente al tipo de dato de cada columna. En esta sección se definirá dicha estructura para el modelo relacional común.

4.1. Definición del modelo relacional por cada fuente de datos

A continuación, se especificarán brevemente el modelo relacional individual por cada fuente de datos.

4.1.1. Base de Datos

El modelo relacional propuesto que representa una base de datos, básicamente tiene la siguiente estructura:

- Clase SCHEMA: Representa la base de datos como tal. Almacenará los metadatos extraídos del schema.
- Clase TABLE: Representa el conjunto de tablas del que está compuesto la base datos. Almacenará los metadatos extraídos de cada tabla.
- Clase COLUMN: Almacenara los metadatos extraídos de cada una de las columnas de las diferentes tablas.
- Clase PRIMARYKEY: Representa las columnas consideradas como parte de la clave primaria de la tabla. Puede ser considerada como una generalización de la clase COLUMN.
- Clase FOREIGNKEY: Representa las columnas consideradas como clave foránea o vínculo con los datos de otra tabla. Es considerada como una generalización de la clase COLUMN

Una observación importante en base al modelo ontológico común descrito anteriormente, es que clases como PRIMARYKEY o FOREIGNKEY no son importantes para la finalidad de este trabajo, puesto que no son consideradas en el modelo ontológico. Razón por la cual este trabajo, el modelo se simplificaría únicamente a tres clases principales básicas SCHEMA, TABLE y COLUMN (figura 4.1)

Tomando como referencia las propiedades de cada una de las clases del modelo ontológico común, se procedió a definir los atributos de las diferentes clases del modelo relacional de la base de datos. Obteniendo el siguiente modelo que representa la estructura de esta fuente de datos. (figura 4.2).

4.1.2. CSV

En el modelo que se propone, un archivo CSV puede ser considerado como una única tabla, la misma que estará formada por columnas que representan los datos separados por comas.

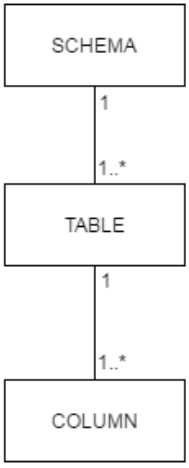


Figura 4.1: Modelo relacional propuesto Base Datos

Por lo tanto un archivo CSV se puede definir como un dataset, donde los datos son almacenados en una única tabla, que a su vez contiene varias columnas. El modelo relacional se podría resumir en las siguientes clases:

- Clase DATASET CSV: representa el archivo CSV propiamente.
- Clase TABLE CSV: representa la única tabla en la que se almacena los diferentes datos
- Clase COLUMN CSV: representa las diferentes columnas en la que se distribuyen los datos, estas columnas son obtenidos mediante la separación por comas de los diferentes registros.

Una consideración importante en un archivo CSV es que al tratarse de un dataset formado por una sola tabla los metadatos o atributos de la clase DATASET CSV y TABLE CSV en su gran mayoría son los mismos. Sin embargo, se dividen en dos clases diferentes, para poder formar un modelo relacional que tenga la misma estructura básica que el resto de modelos relacionales y facilitar su integración en un solo modelo general.

Tomando como referencia las propiedades de cada una de las clases del modelo ontológico común, se procedió a definir los atributos de las diferentes clases del modelo relacional de un archivo CSV. El modelo relacional final correspondiente a un archivo CSV se puede observar el la figura 4.3

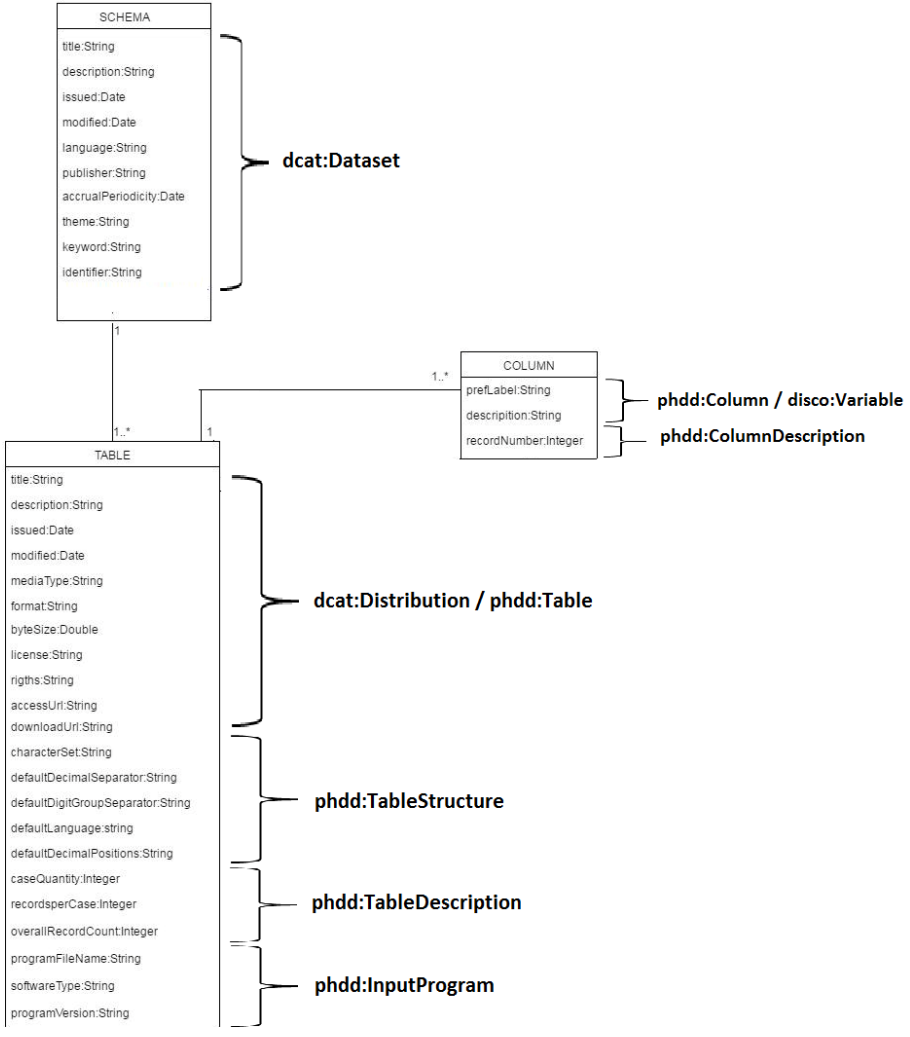


Figura 4.2: Modelo relacional propuesto para una Base de datos (atributos y clases)

4.1.3. Excel

El modelo relacional propuesto, que representa la estructura mediante la cual, un archivo Excel almacena la información de texto plano, puede considerar las siguientes clases:

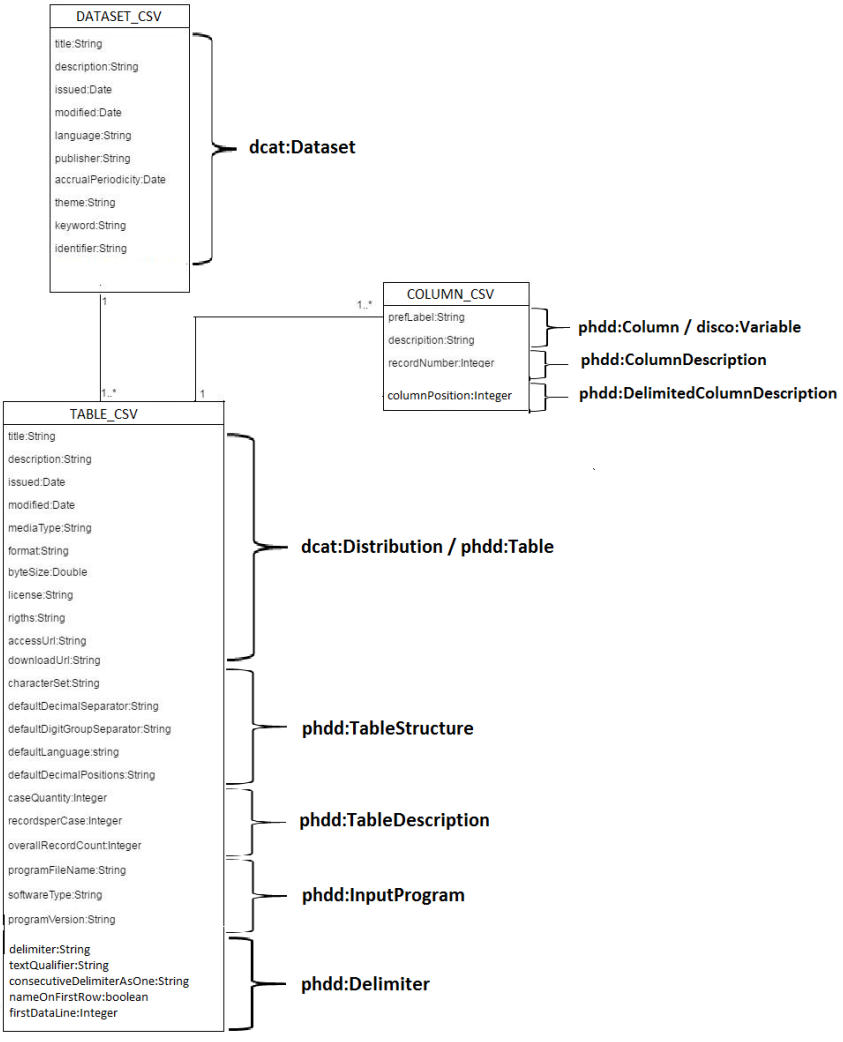


Figura 4.3: Modelo relacional propuesto CSV (atributos y clases)

- Clase WorkBook Representa el conjunto de datos en si.
- Clase WorkSheet Representa las diferentes paginas o hojas con las que cuenta un workbook, es decir archivos de texto plano (Tablas).
- Clase Column_Excel Representa cada una de las columnas que contiene una pagina o worksheet del archivo Excel

Haciendo una analogía con las fuentes de datos anteriores, un archivo Excel puede ser considerado como un conjunto de datos (WORKBOOK), el mismo que se compone de una o mas tablas (WORKSHEET) donde se almacena los datos. Cada tabla puede contener una o mas columnas y filas, en las columnas se almacenara información sobre cada elemento mientras que una fila representa un registro.

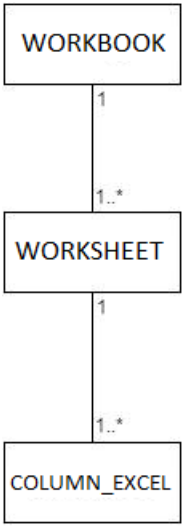


Figura 4.4: Modelo relacional propuesto EXCEL

4.1.4. XML

Para poder definir el modelo relacional de un archivo XML, primero se debe considerar ciertos aspectos acerca del archivo y como se interpretara su estructura para formar el modelo relacional.

- Un archivo XML está formado básicamente por nodos o elementos, que a su vez contienen otros elementos y atributos. Los diferentes elementos constan de un único elemento raíz el cual se caracteriza por ser el único elemento que no tiene un elemento padre.
- Cada nodo o elemento adicionalmente puede contener atributos, es decir un nodo puede contener tanto atributos como nodos hijos.
- Un documento XML debe cumplir con los siguientes requisitos:

- El documento debe tener un solo elemento raíz.
- Todas las etiquetas (tags) abiertas deben tener su respectivas etiquetas de cierre.
- XML distingue mayúsculas de minúsculas por lo que todos los elementos y atributos deben seguir la definición.
- Todos los elementos deben estar correctamente anidados.
- Los valores de los atributos deben ir entre comas simples o dobles.
- No se pueden repetir atributos en un mismo elemento. Por ejemplo, si se quiere representar múltiples autores para un libro, se debe definir el autor como un elemento y no como un atributo.

En la Figura 4.5 se puede observar gráficamente la estructura de un archivo XML.

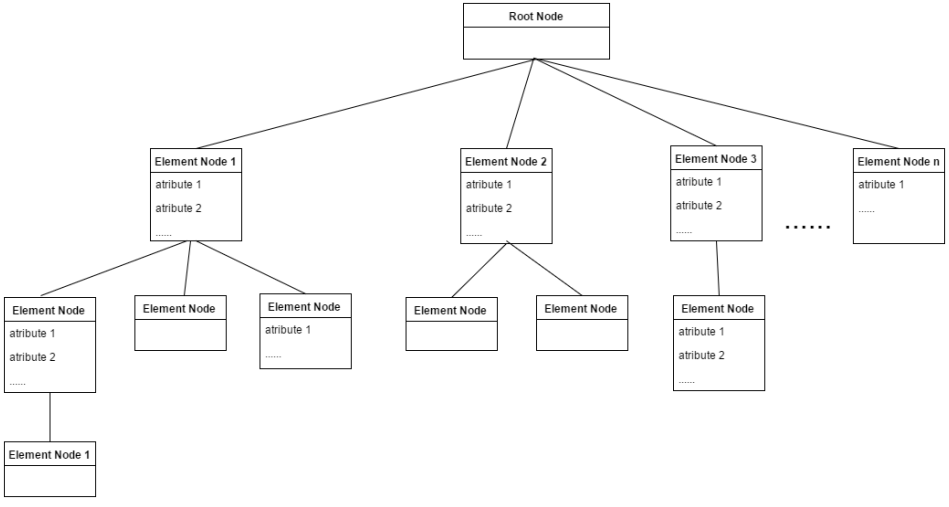


Figura 4.5: Estructura archivo XML

Para el propósito de construir el modelo relacional del archivo XML, los diferentes elementos (nodos) de la estructura, recibirán el siguiente tratamiento.

- Nodo RAÍZ será considerado como el dataset, cuyos metadatos se obtendrán del prologo del archivo XML.
- Nodos o elementos los cuales contenga nodos hijos serán considerados como archivos de formato rectangular (Tablas).

- Nodos o elementos los cuales contenga atributos serán considerados como archivos de formato rectangular (Tablas).
- Nodos o elementos los cuales no contengan nodos hijos y tampoco contengan atributos serán considerados como columnas de su nodo padre.
- Atributos de los nodos o elementos serán considerados como columnas del nodo.

Con estas consideraciones, se puede definir que un elemento o nodo será considerado como tabla cuando tenga al menos un atributo o un elemento hijo. Los atributos y los nodos hijos que a su vez no contenga otros nodos hijos serán considerados como columnas.

En la figura 4.6 se observa el tratamiento que se realizara sobre la estructura del archivo XML a fin de obtener su respectivo modelo relacional.

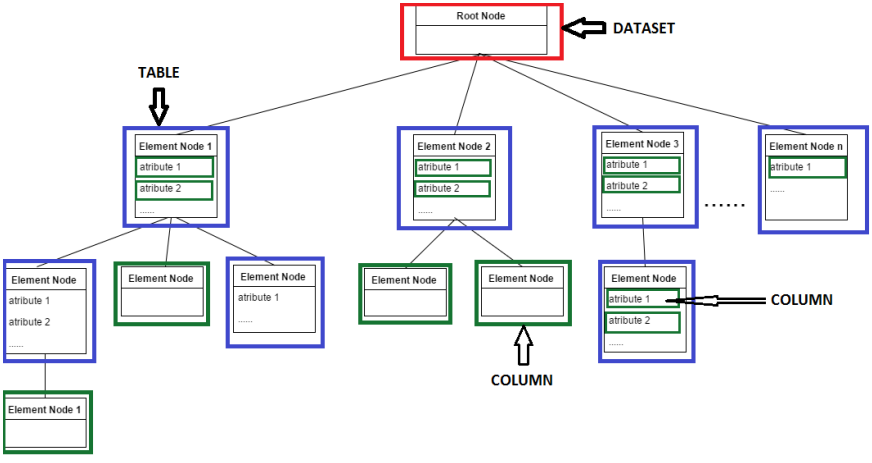


Figura 4.6: Tratamientos sobre la estructura del archivo XML

Una vez definido como se manipulará la estructura del archivo XML, se deduce que los datos se distribuirán al igual que las fuentes de datos anteriores en una estructura formada básicamente por tablas y columnas(Figura 4.7). Por lo que, la estructura que se propone para el modelo relacional, puede considerar las siguientes clases:

- DATASET_XML: Representa el nodo raíz del archivo XML (Dataset)

- TABLE_XML: Representa un nodo el cual contenga por lo menos un atributo o un nodo hijo que no es a su vez nodo padre (Tablas).
- COLUMN_XML: Representa los atributos y nodos hijos que no son nodos padres, de un nodo o elemento (Column).

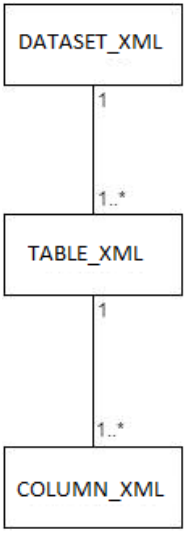


Figura 4.7: Modelo relacional propuesto XML

4.2. Definición del modelo relacional común

Una vez que se han definido el modelo relacional por cada fuente de datos, se debe unir estos con el fin de obtener un modelo relacional común para las distintas fuentes de datos.

Analizando los diferentes modelos de las fuentes de datos se concluye que:

- Las clases:
  - SCHEMA (modelo relacional de una base de datos)
  - DATASET_CSV (Modelo relacional de un archivo CSV)
  - WORKBOOK (Modelo relacional de un archivo EXCEL)
  - DATASET_XML (Modelo relacional de un archivo XML)

Contienen exactamente los mismos atributos independientemente si estos se extraen de forma automática o manual, por lo que estas clase se pueden definir en una sola clase, la cual definirá como DATASET.

- Las clases:
  - TABLE (modelo relacional de una base de datos)
  - TABLE_CSV (Modelo relacional de un archivo CSV)
  - WORKSHEET (Modelo relación de un archivo EXCEL)
  - TABLE_XML (Modelo relacional de un archivo XML)

  Contienen algunos atributos en común, mientras que otros atributos son específicos de cada modelo por lo que para estas clases se procederá a crear una clase general denominada TABLE, y se especializarán, las clases que permitan definir metadatos propios de cada fuente, las clases que heredan de TABLE se denominarán:
  - TABLE_BD: Representa un archivo formato rectangular de una base de datos
  - TABLE_CSV: Representa un archivo formato rectangular de un archivo CSV
  - TABLE_XML: Representa un archivo formato rectangular de un archivo XML
  - TABLE_EXCEL: Representa un archivo formato rectangular de un archivo EXCEL

- Las clases:
  - COLUMN (Modelo relacional de una base de datos)
  - COLUMN_CSV (Modelo relacional de un archivo CSV)
  - COLUMN_EXCEL (Modelo relación de un archivo EXCEL)
  - COLUMN_XML (Modelo relacional de un archivo XML)

  Contienen algunos atributos en común, mientras que otros atributos son específicos de cada modelo por lo cual para estas clase se procederá a crear una clase general denominada COLUMN, y se especializaran de la misma clases que nos permitan definir metadatos propios de cada fuente, las clases que heredan de COLUMN se denominaran:
  - COLUMN_BD: Representa las columnas de una tabla de una base de datos
  - COLUMN_CSV: Representa las columnas de un archivo formato rectangular de un archivo CSV

  - COLUMN_XML: Representa las columnas de una tabla del archivo XML
  - COLUMN_EXCEL: Representa las columnas de una tabla del archivo EXCEL

- Adicionalmente para poder definir el tipo de dato de cada columna en los diferentes modelos se incluye la estructura StorageFormat. Por lo que en el modelo relacional común se deberá especificar dicha estructura sin importar el tipo de fuente que se trate, por lo que esta se relaciona directamente con la clase COLUMN

4.3. Definición de la estructura 'StorageFormat'

Una consideración importante sobre el modelo relacional obtenido, es el tipo de dato de una columna, como se especificó en el capítulo tres, una modificación importante sobre los vocabularios seleccionados es la posibilidad de especificar los distintos tipos de datos, en los que los datos de cada columna pueden ser convertido. Es así, que en el modelo relacional se debe incluir la estructura que permita esta especificación.

La obtención del tipo de dato de una columna, difiere entre los distintas fuentes de datos. Si bien en una base de datos se puede obtener el tipo de dato de una columna con una veracidad del cien por ciento al extraer la información directamente desde el diccionario de datos. En otras fuentes como EXCEL, CSV y XML, el proceso para determinar el tipo de columna incluye analizar los datos del dataset.

Para el análisis de los datos del dataset, se tomará aleatoriamente un porcentaje del total de registros, porcentaje que puede ser considerado como variable y que puede ser ingresado directamente por el encargado de la generación del RDF sobre el modelo ontológico común.

Para poder especificar los distintos tipos de datos de una columna se propone la siguiente estructura:

- Clase StorageFormat: permitirá definir que porcentaje, del total de registros analizados del dataset, son considerados con un determinado tipo de dato. Por ejemplo, de un archivo de 100 registros para determinar el tipo de dato de una columna X, se decidió tomar aleatoriamente y analizar el 70 por ciento (70 registros) obteniendo la siguiente información.
  - 30 % de los 70 registros analizados son de tipo Integer.

- 60 % de los 70 registros analizados son de tipo Decimal.
- 10 % de los 70 registros analizados son de tipo String.

El análisis de datos para obtener el tipo de la columna en una base de datos no es necesario, puesto que el tipo de dato de cada columna se puede extraer directamente del diccionario de datos. Sin embargo en otras fuentes descritas posteriormente como CSV, Excel, etc. Este análisis es sumamente importante para determinar el tipo de dato de sus columnas.

Estos resultados puede ayudar al encargado de la integración entre otras cosas a identificar la presencia de ruido en los datos de la columna ya que existe inconsistencias en el tipo de dato de la misma. Reflejándose de mayor manera por ese 10 % de datos que son de tipo String sobre el 90 % que son de tipo numérico.

- Adicionalmente, cada tipo de dato se podría especificar más a detalle, con atributos específicos, por lo que se decidió crear diferentes clases que representen los diferentes tipos de datos existentes. Las mismas serán una especialización de la clase StorageFormat, entre las clases que se identificaron están las siguientes:

  - Clase String: Que representa las columnas que son de tipo cadena.
  - Clase Integer: Que representa las columnas que son de tipo numérico y pertenecen al conjunto de los números enteros.
  - Clase Decimal: Que representa las columnas que son de tipo numérico y pertenecen al conjunto de los números racionales.
  - Clase Date: Que representa las columnas que son o pueden ser consideradas como fechas.
  - Clase Boolean: representa las columnas que son de tipo booleano.

  Los atributos específicos de cada tipo de dato, se puede observar en la tabla 3.1

- El porcentaje de datos que se analizarán para obtener los tipos de datos de una columna, será parametrizable y deberá ser ingresado por el usuario encargado de la generación del RDF sobre el modelo ontológico común. Este porcentaje será definido como un atributo más de la clase DATASET, ya que es común para todas las tablas y columnas de las diferentes fuentes de datos, es decir si el usuario especifica el 70 % de los datos, en cada columna de cada tabla, se analizarán este porcentaje de los datos que contengan.

En Figura 4.8 se observa el modelo relacional común para las distintas fuentes de datos, incluyendo la estructura para el almacenamiento del tipo de dato de cada columna (StorageFormat).

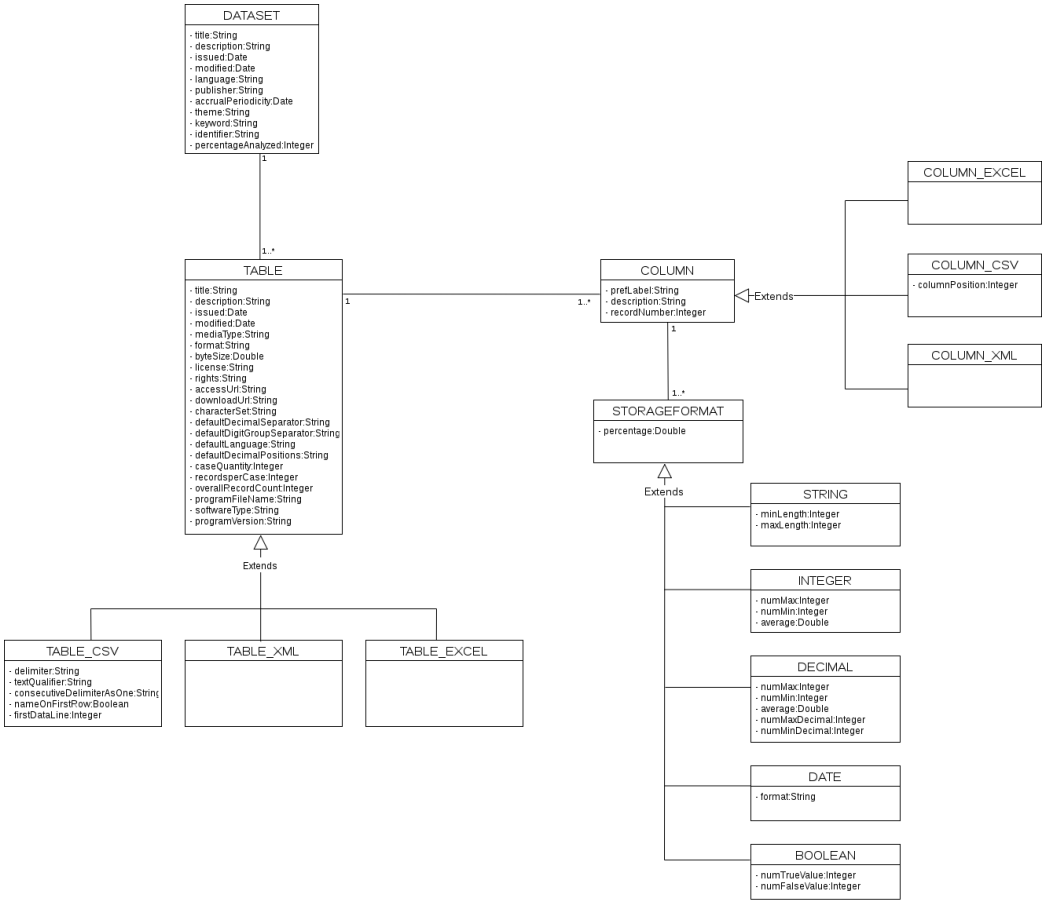


Figura 4.8: Modelo relacional general

Capítulo 5

Proceso de extracción y almacenamiento temporal de metadatos

En este capítulo, se especificará el proceso de extracción de los metadatos de cada fuente de datos y mapeo entre estos metadatos extraídos y el modelo relacional común, definido en el capítulo cuatro. Se describirán los procesos tanto para una fuente de datos de tipo CSV como para una Base de Datos.

5.1. Tratamiento de fuente de datos CSV

En el tratamiento para una fuente de datos CSV, primero se realiza el proceso de extracción de los metadatos y posteriormente se realiza el mapeo de los metadatos extraídos con el modelo relacional común.

5.1.1. Proceso de extracción de metadatos

En la figura 4.3 se especifica el modelo de un archivo CSV, en este, al igual que en otras fuentes de datos, existen atributos como *keyword*, *contactpoint*, *theme* entre otros, los cuales no se pueden obtener mediante procesos automáticos ya que no se dispone de herramientas o procesos para extraer esta información o simplemente en la creación de la fuente no fueron especificados dichos metadatos.

Adicionalmente, habrá atributos que no se podrán extraer directamente, por lo que se deberán crear procesos programados o realizar análisis de los datos de la fuente para su obtención.

Para la extracción de los metadatos correspondientes a un archivo CSV, ha diferencia de una base de datos, no dispone de una estructura que proporcione directamente los metadatos de la fuente. Por lo que se hará uso de diferentes procesos y herramientas para dicha extracción.

Entre las herramientas seleccionadas para la extracción, se encuentran los siguientes:

- **APACHE TIKA[18]:** Es un conjunto de herramientas para extraer contenido y metadatos de muchos formatos de archivo diferentes como MS Word, Excel o CSV, que son difíciles de analizar sin las bibliotecas adecuadas.

- **Paquete JAVA NIO[19]:** Disponible en java 7, fue añadido para soportar I/O mapeada en memoria, facilitando las operaciones I/O cercanas al hardware subyacente con mejor rendimiento.

- **Paquete JAVA IO [20]:** El paquete java.io contiene clases que soportan entrada/salida. Las clases del paquete son principalmente streams; sin embargo, se incluye una clase para ficheros de acceso aleatorio.

Para la obtención de metadatos o atributos definidos en la figura 4.3, es importante destacar la manera de cómo estos atributos van a ser extraídos, de forma automática, manual o mediante procesos de análisis de datos.

Extracción automática	**Registro o extracción manual**	**Generados mediante procesos y análisis de datos**
title	publisher	identifier
issued	accrualPeriodicity	
modified	theme	
language	keyword	
modified	contactPoint	
description		

Cuadro 5.1: Formas de extracción metadatos para la clase DATASET - CSV.

En el cuadro 5.1 se indica las respectivas formas de extracción de metadatos para la clase DATASET_CSV

Extracción automática	Registro o extracción manual	Generados mediante procesos y análisis de datos
title	license	defaultDecimalSeparator
	rigths	defaultDigitGroupSeparator
issued	theme	defaultDecimalPositions
modified	accessUrl	programVersion
mediaType	downloadURL	programFileName
format	newLine	softwareType
bytesize	caseQuantity	consecutiveDelimiterAsOne
characterSet	recordPerCase	nameOnFirstRow
defaultLanguage	description	firstDataLine
defaultLocale		
overallRecordCount		
Delimiter		
textQualifier		

Cuadro 5.2: Forma extracción metadatos clase TABLE_CSV

En el cuadro 5.2 se indica las respectivas formas de extracción de metadatos para la clase TABLE_CSV

Extracción automática	Registro o extracción manual	Generados mediante procesos y análisis de datos
prefLabel	description	recordNumber
		columnPosition

Cuadro 5.3: Forma extracción metadatos clase COLUMN_CSV

En el cuadro 5.3 se indica las respectivas formas de extracción de metadatos para la clase COLUMN_CSV

5.1.2. Mapeo entre los metadatos de la fuente con el modelo relacional común

Una vez extraídos los diferentes metadatos del archivo CSV, estos deben ser almacenados de forma temporal en el modelo relacional, como paso previo a la generación del RDF. Dicho almacenamiento requiere de un mapeo entre los metadatos extraídos y los atributos de dicho modelo relacional.

En el cuadro 5.4 se resumen los metadatos extraídos de un archivo CSV y su mapeo con el modelo relacional específicamente con la clase DATASET, adicionalmente se especifican los procesos y herramientas utilizadas para dicha extracción.

Atributo Modelo Relacional	Herramienta	Proceso/Método
title	java.io.File	getName()
issued	java.nio.file	creationTime()
modified	BasicFileAttributes	lastModifiedTime()
language	java.nio.file	metadata.get(Çontent-Language”);
publisher	BasicFileAttributes	Extracciòn manual
accrualPeriodicity	org.apache.tika	Extracciòn manual
theme		Extracciòn manual
keyword		Extracciòn manual
identifier		Proceso programado de generacion de uuid

Cuadro 5.4: Extracción metadatos y mapeo con el modelo relacional (Clase DATASET - CSV)

El mapeo de los metadatos extraídos del archivo CSV y el resto de clases del modelo relacional, en este caso TABLE_CSV y COLUMN_CSV, se especifican en el Anexo C.

5.1.3. Ejemplo de tratamiento de fuente datos CSV

En el presente apartado, se describe mediante un ejemplo específico, la extracción de los metadatos así como el mapeo de estos con el modelo relacional, para el tratamiento de una fuente CSV. Esto con la finalidad de que quede mucho más claro el proceso.

Análisis de la fuente de datos

La fuente de datos que se toma como ejemplo, es un conjunto de datos proporcionado por el departamento de la Universidad de Cuenca PROMAS[37].El archivo dispone de la siguiente estructura.

- El archivo describe datos obtenidos acerca del nivel y temperatura del caudal del rio Baro Pachala, en un periodo de tiempo específico.
- El archivo CSV está formado por 10000 registros
- El archivo contiene las siguientes columnas:
  - No. Identificador único de cada registro del *dataset*
  - *Date/Time* Representa el periodo de tiempo (fecha), en el que fue recogido la muestra.

- *Level* Representa el nivel de caudal del río, que se recogió para la muestra.
- *Temperature* Representa la temperatura del río, que se recogió para la muestra.

- Una consideración importante sobre el *dataset* es la incorporación de ruido, ya que para sacar el máximo potencial del modelo ontológico común propuesto, se deben tratar con datos heterogéneos. Es así, que en cada columna se incorporó caracteres adicionales sobre los datos, con la finalidad de observar el desempeño del modelo en el peor de los casos. En la figura 5.1, se puede observar una parte de los datos, con los que se trabajarán.

```
No.,Date/time,Level[cm],Temperature [ºC];;;;
1,2015-07-09 11:15:00,805.1,29.73;;;;
2,2015-07-09 11:30:00,805.1,27.42;;;;
3a,2015-07-09 11:45:00,805.1,25.31;;;;
4,2015a-07-09 12:00:00, ,23;;;;
5#,2015-07-09 12:15:00, ,21;;;;
6,2015/07/09 12:30:00,804.9,21.26000000;;;;
7,2015-07-09 12:45:00,804$.9,20.0056;;;;
8,2015/07/09 13:00:00,804.7,20.64;;;;
null,2015-07-09 13:15:00,804.6055454556,21.59;;;;
```

Figura 5.1: Archivo CSV utilizado como ejemplo

Extracción de los metadatos de la fuente y mapeo con el modelo relacional común

El modelo relacional común, que almacenará temporalmente los metadatos extraídos de las diferentes fuentes se puede observar en la figura 4.8, mientras que en la figura 4.3 se describe más a detalle la estructura o modelo relacional correspondiente a un archivo CSV.

En las tablas 5.5, 5.6, 5.7, 5.8, 5.9, 5.10 se muestra los principales metadatos obtenidos del archivo CSV, los procesos y herramientas que fueron utilizadas para la extracción y el posterior mapeo con el modelo relacional general, en específico con la clase DATASET.

Atributo Modelo Relacional	**Herramienta**	**Proceso/Método**	**Metadato Extraído**
title	java.io.File	**getName()** Metodo que retorna el nombre del archivo o directorio	Caudal dive Baro Pacchala291115
description		**Extracción manual** Con la finalidad de describir de la mejor manera el dataset, se ingresó de forma manual la descripción.	"El siguiente dataset, describe los datos recogidos durante del años 2015 y 2016 acerca de los niveles y la temperatura del caudal del río diver Baro"
issued	java.nio.file.attribute. BasicFileAttributes	**creationTime()** Método que retorna la fecha en la que el archivo fue creado	"02 de octubre de 2015, 22:02:01"
modified	java.nio.file.attribute. BasicFileAttributes	**lastModifiedTime()** Retorna la fecha de la última modificación del archivo.	"30 de octubre de 2015, 13:02:01"
language	org.apache.tika	**metadata.get(Content-Language)** Retorna el lenguaje de una pieza del contenido del archivo.	"es"
theme		**Extracción manual**	"Descripción de fuentes hidrológicas"
keyword	java.io.File	**Extracción manual** Con la finalidad de describir de la mejor manera el dataset, se ingresó de forma manual el metadato	"caudal" "river" "temperature" "nivel" "csv"
identifier	java.util.UUID	**ramdomUUID()** Retorna un identificador único de tipo 4 (pseudo randomly generated).	"067e6162-3b6f-4ae2-a171-2470b63dff00"

Cuadro 5.5: Ejemplo de extracción de atributos/metadatos y mapeo con el modelo relacional. DATASET- CSV

Atributo Modelo Relacional	**Herramienta**	**Proceso/Método**	**Metadato Extraído**
title	java.io.File	**getName()** Metodo que retorna el nombre del archivo o directorio	Caudal_diver_Baro Pacchala291115
description		Extracción manual Con la finalidad de describir de la mejor manera el dataset, se ingresó de forma manual la descripción.	"El siguiente dataset, describe los datos recogidos durante del años 2015 y 2016 acerca de los niveles y la temperatura del caudal del río diver Baro"
issued	java.nio.file.attribute. Basic File Attributes	creationTime() Método que retorna la fecha en la que el archivo fue creado	"02 de octubre de 2015, 22:02:01"
modified	java.nio. file. attribute. BasicFileAttributes	lastModifiedTime() Retorna la fecha de la última modificación del archivo.	"30 de octubre de 2015, 13:02:01"
license	org.apache.tika	Extracción manual	
rigths		Extracción manual	
accessUrl		Extracción manual	
download Url		Extracción manual	
mediaType		Extracción manual	aplication/csv
format		Extracción manual	aplication/csv
byteSize	java.io.File	length()	453.428 bytes
character Set	org.apa che.tika	metadata.get(Çontent-Type")	
default Language	org.apa che.tika	metadata.get(Çontent-Language")	es
default Locale	org.apa che.tika	metadata.get(Çontent-Type")	es
newLine		Extracción manual	
case Quantity		Extracción manual	
recordPer Case		Extracción manual	
overrall Record Count	org.apache.tika	readRecord()	10045
delimiter	com.csv reader. Csv Reader	getDelimiter()	","

text Qualifier	com.csv reader. Csv Reader	gettextQualifier()	(‘’)
first Data Line		1	2
consecutive Deli miterAsOne			
nameOn First Row			true

Cuadro 5.6: Ejemplo de extracción de atributos/metadatos y mapeo con el modelo relacional. TABLE_CSV

Atributo Modelo Relacional	**Herramienta**	**Proceso/Método**	**Metadato Extraído**
prefLabel	com.csv reader. Csv Reader	getHeaders()	No.
description		Extracción manual	
column Position	com.csv reader. Csv Reader	getHeaders()	0
record Number	com.csv reader. Csv Reader	readRecord()	10045

Cuadro 5.7: Ejemplo de extracción de atributos/metadatos y mapeo con el modelo relacional COLUMN_CSV (No.)

Atributo Modelo Relacional	**Herramienta**	**Proceso/Método**	**Metadato Extraído**
prefLabel	com.csv reader. Csv Reader	getHeaders()	Date/time.
description		Extracción manual	
column Position	com.csv reader. Csv Reader	getHeaders()	1
record Number	com.csv reader. Csv Reader	readRecord()	9875

Cuadro 5.8: Ejemplo de extracción de atributos/metadatos y mapeo con el modelo relacional COLUMN_CSV (Date/time)

Atributo Modelo Relacional	**Herramienta**	**Proceso/Método**	**Metadato Extraído**
prefLabel	com.csv reader. Csv Reader	getHeaders()	Level[cm]
description		Extracción manual	
column Position	com.csv reader. Csv Reader	getHeaders()	2
record Number	com.csv reader. Csv Reader	readRecord()	10045

Cuadro 5.9: Ejemplo de extracción de atributos/metadatos y mapeo con el modelo relacional COLUMN_CSV (Level[cm])

Atributo Modelo Relacional	**Herramienta**	**Proceso/Método**	**Metadato Extraído**
prefLabel	com.csv reader. Csv Reader	getHeaders()	Temperature [°C]
description		Extracción manual	
column Position	com.csv reader. Csv Reader	getHeaders()	3
record Number	com.csv reader. Csv Reader	readRecord()	10036

Cuadro 5.10: Ejemplo de extracción de atributos/metadatos y mapeo con el modelo relacional COLUMN_CSV (Temperature [°C])

Extracción metadatos correspondiente a la estructura StorageFormat

Como se mencionó anteriormente, una consideración importante sobre el modelo relacional general, así como sobre el modelo ontológico común. Es la estructura que almacenará los metadatos correspondientes al tipo de dato que cada columna (StorageFormat).

Para la obtención de los metadatos de dicha estructura se tomó en cuenta las siguientes consideraciones:

- El archivo CSV está formado por 10000 registros
- Para la obtención del tipo de dato de cada columna, se trabajará con una muestra que represente el noventa por ciento (90 %) del total de registros del archivo.

Para determinar el tipo de dato de la columna, se tomará aleatoriamente el porcentaje especificado por el usuario del total de registros. De cada registro se obtendrán los datos de la columna que se desea analizar y se tratarán de convertir a diferentes formatos. Se registrará el porcentaje de los datos que fueron convertidos exitosamente a los diferentes tipos de formato. Adicionalmente, dependiendo del tipo de dato que se obtenga de cada columna, se especificarán los atributos propios, por ejemplo al tratarse de una cadena, se deberán especificar la longitud de la cadena más larga y de igual manera la más corta.

En los cuadros 5.11, 5.12, 5.13 y 5.14. Se especifican los tipos de dato de cada columna que se obtuvieron analizando el 90 % del total de registros.

Tipo de dato	Porcentaje	Atributos Específicos
INTEGER	83 %	numMin: 1 y numMax: 9456
STRING	17 %	minLength: 2 y maxLength: 4

Cuadro 5.11: Tipo de datos correspondientes a la columna No.

Tipo de dato	Porcentaje	Atributos Específicos
DATE	71 %	format: YYYY-MM-DD HH:MM:SS
DATE	21 %	format: YYYY/MM/DD
DATE	8 %	format: YYYY-MM-DD HH:MM:SS.zzz

Cuadro 5.12: Tipo de datos correspondientes a la columna Date/Time

Tipo de dato	Porcentaje	Atributos Específicos
DECIMAL	92 %	▪ minValue: 801.1 ▪ maxValue: 807.7 ▪ minDecimalPositions: 1 ▪ minDecimalPositions: 7
INTEGER	2 %	▪ minValue: 801 ▪ maxValue: 806
STRING	6 %	▪ minLength: 5 ▪ maxLength: 6

Cuadro 5.13: Tipo de datos correspondientes a la columna Temperature.

Tipo de dato	Porcentaje	Atributos Específicos
DECIMAL	97 %	▪ minValue: 19.8 ▪ maxValue: 33.6 ▪ minDecimalPositions: 2 ▪ minDecimalPositions: 9
STRING	3 %	▪ minValue: 801 ▪ maxValue: 806
STRING	6 %	▪ minLength: 5 ▪ maxLength: 6

Cuadro 5.14: Tipo de datos correspondientes a la columna Level.

El cuadro 5.13 describe el tipo de dato de la columna temperature, la cual está representada por el tipo Decimal que es el 92 %. Pero es importante destacar que existe ruido, ya que hay el 2 % de datos representados con el tipo Integer y un 6 % de datos representados con el tipo String. En definitiva, la columna Temperature sera representado como Decimal.

5.2. Tratamiento de fuente de datos Base De Datos

Para el caso de una base de datos, al igual que en un archivo CSV se sigue el mismo proceso de extracción de metadatos y el mapeo de estos, sobre el modelo ontológico común.

5.2.1. Proceso de extracción de metadatos

En el caso de una base de datos los atributos como *keyword*, *contactpoint*, *theme*, entre otros, es imposible obtenerlos mediante procesos automáticos, ya que la estructura que nos provee los metadatos, en este caso el diccionario de datos, no almacena información referente a estos atributos. Adicionalmente existirán atributos los cuales serán generados mediante procesos programados, caso concreto *identifier*, mientras que para la obtención de otros metadatos será necesario analizar los datos en si del *dataset*, por ejemplo para determinar el tipo de dato de una columna en ocasiones, se tendrá que analizar los diferentes registros de la fuente de datos e ir tratando de convertir cada uno de estos a diferentes tipos de datos,

a fin de especificar el tipo de dato correcto.

El cuadro 5.15 presenta aquellos metadatos o atributos, que van a poder ser extraídos automáticamente de una base de datos y cuales se deberán ingresar de forma manual.

Extracción automática	**Registro o extracción manual**	**Generados mediante procesos y análisis de datos**
title	publisher	identifier
issued	accrualPeriodicity	
modified	theme	
	keyword	
	contactPoint	
	language	
	description	

Cuadro 5.15: Forma extracción metadatos clase DATASET - Base Datos

De manera similar que la definición de atributos de la clase DATASET, a continuación el cuadro 5.16, se especifica los atributos que pueden ser obtenidos de forma automática y de forma manual para la clase TABLE, mientras que en el cuadro 5.17 se especifica la forma de extracción de metadatos para la clase COLUMN.

Extracción automática	**Registro o extracción manual**	**Generados mediante procesos y análisis de datos**
title	license	defaultDecimalSeparator
description	rigths	defaultDigitGroupSeparator
issued	theme	defaultDecimalPositions
modified	accessUrl	programVersion
mediaType	downloadURL	programFileName
format	newLine	softwareType
bytesize	caseQuantity	
characterSet	recordPerCase	
defaultLanguage		
defaultLocale		
overallRecordCount		

Cuadro 5.16: Forma extracción metadatos clase TABLE - Base Datos

Extracción automática	Registro o extracción manual	Generados mediante procesos y análisis de datos
prefLabel	description	
recordNumber		

Cuadro 5.17: Forma extracción metadatos clase COLUMN - Base Datos

5.2.2. Mapeo entre los metadatos de la fuente con el modelo relacional común

Los metadatos correspondientes a una base de datos, se obtendrán directamente del diccionario de datos, mediante consultas SQL sobre las tablas que conforman esta estructura.

Una vez extraídos los diferentes metadatos de la base de datos, estos deben ser almacenados de forma temporal en el modelo relacional para su tratamiento previo a la generación del RDF, dicho almacenamiento requiere de un mapeo entre los metadatos extraídos y los atributos de dicho modelo.

En el caso de una base MySQL los metadatos son provistos mediante una conjunto de tablas denominado INFORMATION_SCHEMA, esta estructura es una base de datos que almacena información, acerca de todas las otras bases de datos que mantiene el servidor MySQL. Cada usuario MySQL tiene derecho a acceder a estas tablas, pero sólo a los registros que se corresponden a los objetos a los que tiene permiso de acceso.

En los cuadros 5.18, 5.19 y 5.20 se resumen los metadatos extraídos de una base de datos, en este caso MYSQL, los procesos utilizados para la extracción y el mapeo correspondiente entre estos metadatos extraídos y los atributos del modelo relacional.

Atributo Modelo Relacional	**Proceso**
title	SELECT schema_name FROM INFORMATION_SCHEMA.SCHEMATA
description	Extracción manual
issued	SELECT MIN(create_time) FROM INFORMATION_SCHEMA.TABLES where table_schema =
modified	SELECT MIN(update_time) FROM INFORMATION_SCHEMA.TABLES where table_schema =
language	SELECT default_character_set_name FROM INFORMATION_SCHEMA.SCHEMATA
publisher	Extracción manual
accrualPeriodicity	Extracción manual
theme	Extracción manual
keyword	Extracción manual
identifier	Proceso programado de generación de uuid

Cuadro 5.18: Extracción metadatos y mapeo con el modelo relacional (Clase DATASET)

Atributo Modelo Relacional	**Proceso**
title	SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE SCHEMA_NAME =
description	- SELECT TABLE_COMMENT FROM INFORMATION_SCHEMA.TABLES
issued	SELECT CREATE_TIME FROM INFORMATION_SCHEMA.TABLES WHERE SCHEMA_NAME =
modified	SELECT UPDATE_TIME FROM INFORMATION_SCHEMA.TABLES WHERE SCHEMA_NAME =
mediaType	Default base de datos 'application/sql'
format	Default base de datos 'application/sql'
bytesize	SELECT (DATA_LENGTH+INDEX_LENGTH)/(1024-1024) FROM INFORMATION_SCHEMA.TABLES WHERE SCHEMA_NAME =
license	Extracción manual
rigths	Extracción manual
accessUrl downloadUrl	Extracción manual Extracción manual
characterSet	SELECT CCSA.character_set_name FROM INFORMATION_SCHEMA.TABLES T, INFORMATION_SCHEMA.COLLATION CCSA WHERE T.TABLE_SCHEMA = AND T.TABLE_NAME = AND CCSA.COLLATION_NAME = T.TABLE_COLLATION
defaultDecimalSeparator	Proceso programado mediante análisis de datos
defaultDigitGroupSeparator	Proceso programado mediante análisis de datos
defaultLanguage	Extracción manual
defaultLocale	Extracción manual
defaultDecimalPositions	Proceso programado mediante análisis de datos
caseQuantity	Extracción manual
recordsPerCase	Extracción manual
overallRecordCount	SELECT TABLE_ROWS FROM INFORMATION_SCHEMA.TABLES
programFileName	Extracción manual
softwareType	Extracción manual
programVersion	Extracción manual

Cuadro 5.19: Extracción metadatos y mapeo con el modelo relacional (Clase TABLE)

Atributo Modelo Relacional	Proceso
prefLabel	SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME =
recordNumber	SELECT count('COLUMN') FROM SCHEMA.TABLE WHERE 'COLUMN' IS NOT NULL
description	SELECT COLUMN_COMMENT FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME =

Cuadro 5.20: Extracción metadatos y mapeo con el modelo relacional (Clase COLUMN)

5.2.3. Ejemplo de fuente de datos Base de Datos

En el presente apartado, mediante un ejemplo específico se describe el proceso de extracción de metadatos de una base de datos tipo MYSQL y posterior mapeo con el modelo relacional, .

Análisis de la fuente de datos

La fuente de datos que se toma como ejemplo, es una base de datos MYSQL, la misma que almacena información referente a datos climatológicos recogidos durante las últimas dos décadas en diferentes estaciones a lo largo de la provincia del Azuay. Dicho dataset es proporcionado por el departamento de la Universidad de Cuenca PROMAS.

La fuente de datos proporcionada cuenta con una estructura bastante compleja y grande, y siendo el propósito de este apartado describir de forma detallada y puntual el proceso de descripción de la fuente, más no el proceso completo de descripción. Se trabajará únicamente con una tabla y ciertas columnas de la misma, ya que el proceso se repite para las diferentes tablas que contenga la base de datos.

La tabla seleccionada se denomina "clima_diario" y describe datos climatológicos recogidos por diferentes usuarios, en las diversas estaciones climáticas de la provincia en un periodo de tiempo específico.

- La tabla esta formada por 38420 registros
- Las columnas de dicha tabla con las que se procederá a trabajar son:
  - **Idestacion** Identificador único de cada estación, clave foránea que representa la estación en donde se recopiló la muestra de datos.

- **Fecha** Representa el periodo de tiempo (fecha), en el que fue recogido la muestra.
- **Precipitación** Representa el nivel de precipitación de la lluvia recogida para la muestra
- **Nubosidad** Representa el nivel de nubosidad del ambiente recogida para la muestra
- **Evaporación** Representa el nivel de evaporación del ambiente, recogida para la muestra
- **Temperatura** Representa la temperatura del ambiente, recogida para la muestra
- **Viento** Representa la velocidad del viento en el ambiente, recogida para la muestra

*	idestacion	fecha	precipitacion	nubosidad	evaporacion	temperatura	viento
1	1	2014-01-05	548.5	452.47	45.5	12.69	874
2	5003	2014-01-05	784.69	54	69	23	00.3
3	9005	2014-01-05	8874	87	56	19	56
4	5009	2014-12-04	541.25	65	87	12	89
5	874	2014-09-01	547	65	78	6	78.3
6	587	2014-01-09	987	45	98	-8	12.55
7	565	2014-09-05	548.5	45	89	12	45

Figura 5.2: Base de Datos utilizado como ejemplo

Mapeo entre la fuente y el modelo relacional general

El modelo relacional común, que almacenará temporalmente los metadatos extraídos de las diferentes fuentes, se puede observar en la Figura 4.8, mientras que en la Figura 4.2 se describe más a detalle la estructura o modelo relacional correspondiente a una base de datos.

Una vez especificado la estructura o modelo relacional que albergará los metadatos hasta su mapeo con el modelo ontológico común y posterior generación RDF, adicionalmente se describirán los procesos de obtención o extracción de los metadatos de la fuente. Para el caso de una base de datos, los metadatos se extraen directamente desde el diccionario de datos. La forma de obtención de los metadatos (automático/manual) se describió anteriormente.

En los cuadros 5.21, 5.22 y 5.23 muestran los metadatos extraídos para las clases DATASET, TABLE y COLUMN respectivamente, adicionalmente se especifican los procesos y herramientas que fueron utilizadas para la extracción.

Atributo Modelo Relacional	**Proceso**	**Resultado**
title	SELECT schema_name FROM INFORMATION_SCHEMA.SCHEMATA	Clima
description	Extracción manual	
issued	SELECT MIN(create_time) FROM INFORMATION_SCHEMA.TABLES where table_schema =	2014-01-12
modified	SELECT MIN(update_time) FROM INFORMATION_SCHEMA.TABLES where table_schema =	2015-10-22
language	SELECT default_character_set_name FROM INFORMATION_SCHEMA.SCHEMATA	es
publisher		
accrual Periodicity		
theme	Extracción manual	Descripción de fuentes/datos climatológicos
keyword	Extracción manual	“precipitation” “climate” “temperature” “database” “azuay“
identifier	Proceso programado de generación de uuid	076e6162-3b6f-6ae2-n489-2470b63dtf00

Cuadro 5.21: Ejemplo de extracción de atributos/metadato y mapeo con el modelo relacional DATASET - Base de datos

Atributo Modelo Relacional	**Proceso**	**Resultado**
title	SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE SCHEMA_NAME =	clima_diario
description	- SELECT TABLE_COMMENT FROM INFORMATION_SCHEMA.TABLES	
issued	SELECT CREATE_TIME FROM INFORMATION_SCHEMA.TABLES WHERE SCHEMA_NAME =	2014-01-12 1
modified	SELECT UPDATE_TIME FROM INFORMATION_SCHEMA.TABLES WHERE SCHEMA_NAME =	2015-10-22
mediaType	Default base de datos 'application/sql'	application/sql
format	Default base de datos 'application/sql'	application/sql
bytesize	SELECT (DATA_LENGTH+INDEX_LENGTH) / (1024-1024) FROM INFORMATION_SCHEMA.TABLES WHERE SCHEMA_NAME =	4.292.776 bytes
characterSet	SELECT CCSA.character_set_name FROM INFORMATION_SCHEMA.TABLES T, INFORMATION_SCHEMA.COLLATION CCSA WHERE T.TABLE_SCHEMA = AND T.TABLE_NAME = AND CCSA.COLLATION_NAME = T.TABLE_COLLATION	latin1
overallRecord Count	SELECT TABLE_ROWS FROM INFORMATION_SCHEMA.TABLES	38420

Cuadro 5.22: Ejemplo de extracción de atributos/metadato y mapeo con el modelo relacional TABLE - Base de datos

Atributo Modelo Relacional	**Proceso**	**Resultado**
prefLabel	SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME =	id_estacion
record_number	SELECT count('COLUMN') FROM SCHEMA.TABLE WHERE 'COLUMN' IS NOT NULL	38420
description	SELECT COLUMN_COMMENT FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME = and COLUMN_NAME ='	

Cuadro 5.23: Ejemplo de extracción de metadatos y mapeo con el modelo relacional(Columna id_estacion)

En el Anexo D, se especifica la extracción completa de metadatos, de la base de datos de ejemplo.

Extracción metadatos correspondiente a la estructura StorageFormat

A diferencia del archivo CSV tomado como ejemplo anteriormente, en el caso de una base de datos no se tendrán que analizar los datos del dataset a fin de determinar el tipo de dato de cada columna, ya que el diccionario de datos provee toda esta información así como los atributos propios de cada tipo de dato. En consecuencia, para una base de datos no se tendrán que ingresar el porcentaje de datos que se desea analizar, teniendo como default el 100 %.

Adicionalmente, para una columna de una base de datos, no se podrá especificar diferentes tipos de dato por columna, como era el caso del archivo CSV, ya que el tipo de dato que se extrae del diccionario de datos tiene una fiabilidad del 100 %. Por ejemplo si una columna resulta ser de tipo Decimal, esto significa que no existirá ningún dato que pueda presentar ruido o sea tipo String o Date por citar algunos, ya que los datos que fueron ingresados anteriormente en la base de datos, tuvieron que cumplir con las restricciones propias de cada columna.

Los atributos propios de cada tipo de dato, de igual manera que los metadatos anteriores, se obtendrán directamente del diccionario de datos, mediante consultas

SQL sobre este conjunto de tablas.

En el cuadro 5.24 se especifican el tipo de dato extraído para cada columna, así como los atributos propios de cada tipo de dato. Adicionalmente se especificarán los procesos con los que se determinó dicho tipo.

Columna	Tipo Dato	Atributos	Procesos
id_estacion	Integer	percentage:100 % numMin: 1 numMax: 9456	SELECT DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS WHERE COLUMN_NAME = 'id_estacion'
fecha	Date	format: YYYY-MM-DD HH:MM:SS	SELECT DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS WHERE COLUMN_NAME = 'fecha'
precipitacion	Decimal	percentage:100 % numMin: 19.8 numMax: 33.6 numMinDecimal: 2 NumMaxDecimal: 9	SELECT DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS WHERE COLUMN_NAME = 'precipitacion'
nubosidad	Decimal	percentage: 100 % numMin: 0.0 numMax: 11.42 numMinDecimal: 1 NumMaxDecimal: 2	SELECT DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS WHERE COLUMN_NAME = 'nubosidad'
evaporacion	Decimal	percentage: 100 % numMin: 801.1 numMax: 807.7 numMinDecimal: 1 NumMaxDecimal: 7	SELECT DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS WHERE COLUMN_NAME = 'evaporacion'
temperatura	Decimal	percentage: 100 % numMin: 801.1 numMax: 0.985 numMinDecimal: 1 NumMaxDecimal: 3	SELECT DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS WHERE COLUMN_NAME = 'temperatura'
viento	Decimal	percentage: 100 % numMin: 10.55 numMax: 18.48 numMinDecimal: 2 NumMaxDecimal: 2	SELECT DATA_TYPE FROM INFORMATION_SCHEMA.COLUMNS WHERE COLUMN_NAME = 'viento'

Cuadro 5.24: Tipo de dato de las columnas correspondientes a la base de datos de ejemplo.

Capítulo 6

Mapeo entre modelo relacional - modelo ontológico y generación RDF

En el presente capítulo se definirá el proceso de mapeo entre el modelo relacional y el modelo ontológico, además se menciona los criterios tomados en cuenta para la generación del RDF. Es importante destacar que un mapeo se realiza con el objetivo de mostrar cómo se asocian entre sí, esta asociación es definida por el desarrollador quien debe describir los elementos relacionados entre los modelos.

En el capítulo tres, se definió los atributos del modelo relacional de cada fuente, haciendo referencia a los atributos, de las clases que forma el modelo ontológico común, por lo que la relación (mapeo) entre ambos modelos se dan por estos atributos.

Las diferentes propiedades que se agregaron, así como, las que se modificaron sobre el modelo ontológico común llevarán el prefijo sf = Storage Format

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
DATASET		dcat:Dataset	Class
DATASET	title	dct:title	Property
DATASET	description	dct:description	Property
DATASET	issued	dct:issued	Property
DATASET	modified	dct:modified	Property
DATASET	language	dct:language	Property
DATASET	publisher	dct:publisher	Property
DATASET	accrualPeriodicity	dct:accrualPeriodicity	Property
DATASET	keyword	dcat:keyword	Property
DATASET	identifier	dct:identifier	Property
DATASET	percentageAnalized	sf:percentageAnalized	Property

Cuadro 6.1: Mapeo propiedades DATASET - dcat:Dataset

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
TABLE		dcat:Distribution/ phdd:Table	Class
TABLE	title	dct:title	Property
TABLE	description	dct:description	Property
TABLE	issued	dct:issued	Property
TABLE	modified	dct:modified	Property
TABLE	license	dct:license	Property
TABLE	rigths	dct:rigths	Property
TABLE	accessUrl	dcat:accessUrl	Property
TABLE	downloadUrl	dcat:downloadUrl	Property
TABLE	mediaType	dcat:mediaType	Property
TABLE	format	dct:format	Property
TABLE	byteSize	dcat:ByteSize	

Cuadro 6.2: Mapeo propiedades TABLE - dcat:Distribution

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
TABLE_CSV		phdd:Delimited	Class
TABLE_CSV	delimiter	phdd: delimiter	Property
TABLE_CSV	textQualifier	phdd: textQualifier	Property
TABLE_CSV	consecutiveDe limiterAsOne	phdd:consecutiveDe limiterAsOne	Property
TABLE_CSV	nameOnFirst Row	phdd: nameOnFirst Row	Property
TABLE_CSV	firstDataLine	phdd: firstDataLine	Property

Cuadro 6.3: Mapeo propiedades TABLECSV - phdd:Delimited

En las tablas 6.1, 6.2, 6.3, se observa el mapeo entre el modelo relacional y modelo ontológico, específicamente entre:

- Clase DATASET(Modelo Relacional) - Clase dcat:Dataset (Modelo Ontológico común)
- Clase TABLE(Modelo Relacional) - Clase dcat:Distribution (Modelo Ontológico común)
- Clase TABLE_CSV(Modelo Relacional) - Clase phdd:Delimited (Modelo Ontológico común)

El mapeo completo entre el modelo ontológico común y el modelo relacional general, se adjunta en el Anexo E.

6.1. Generación de RDF

En esta sección se especifica la generación de RDF. Esta generación se enfoca en la creación de un archivo que represente el formato RDF,en el cual esta representado los datos en tripletas, las cuales contienen un sujeto, un predicado y un objeto que describen un recurso. Por lo cual este componente crea las tripletas que corresponden a cada uno de los atributos del modelo ontológico.

El proceso de generación de RDF, toma como entrada tanto, el modelo relacional, que almacena temporalmente los metadatos extraídos de cada fuente. Así como el modelo ontológico común, se realiza el mapeo entre ambos y se obtiene como salida el RDF para su publicación y explotación. En la figura 6.1, se observa más detalladamente el proceso, especificando adicionalmente las herramientas utilizadas para dicha generación.

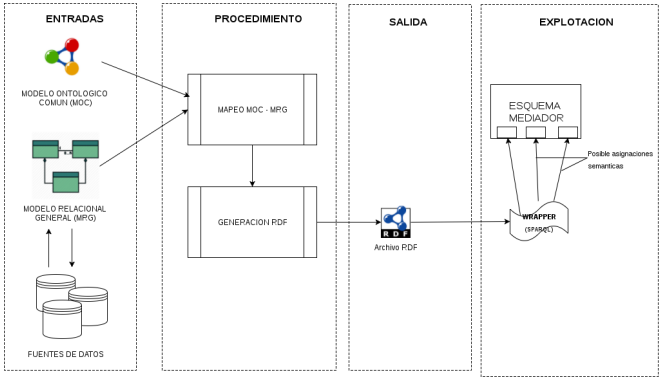


Figura 6.1: Diagrama funcional para la generación de RDF

6.1.1. Ejemplo de Generación de RDF

En el capítulo cinco, se procedió a describir el proceso de extracción de metadatos y el mapeo de estos con el modelo relacional general. En este apartado se mostró ejemplos específicos tanto para un archivo tipo CSV como para un archivo tipo Base de datos. Por lo cual se tomarán estso ejemplos, con el fin de mostrar el mapeo y posterior generación de RDF, entre este modelo relacional generado y el modelo ontológico común, definido en el capítulo dos.

Una consideración importante sobre el modelo ontológico común, es la clase dcat:Catalog, la misma que nos permite básicamente agrupar el conjunto de dataset que se desea describir. Los atributos de esta clase no se pueden definir automáticamente. Ya que la misma guarda información que debe ser ingresada específicamente por el encargado de la descripción de las fuentes de datos, por lo que los atributos de esta clase se definirán manualmente.

En el Anexo F,se define el RDF generado sobre el modelo ontológico común para el caso del conjunto de datos CSV y Base de Datos, utilizando los metadatos extraídos de la fuente y almacenados temporalmente en el modelo relacional común.

Capítulo 7

Aplicación del modelo ontológico común

En este capítulo se muestran los resultados obtenidos con la finalidad de cumplir cada uno de los objetivos planteados al inicio del presente trabajo. En este se ha desarrollado un modelo ontológico común, el mismo que almacena los metadatos de diferente fuentes de datos. Específicamente se trabajó con archivos de base de datos, archivos separados por comas (CSV), archivos XML y archivos Excel. Se contempló archivos de tipo ShapeFile, sin embargo estos almacenan su información en formato CSV, por lo cual fueron considerados con este formato.

La creación de este modelo ontológico se dividió en varias etapas. En una primera etapa se generó el modelo ontológico común, analizando y seleccionando diferentes ontologías de descripción de metadatos. En una segunda etapa se definió un modelo relacional, el mismo que sirve como almacenamiento temporal de los metadatos, previo a la generación del RDF. En la tercera etapa se procedió a extraer los metadatos ya sea de forma automática mediante diferentes herramientas o de forma manual, así como el mapeo entre estos metadatos y el modelo relacional. Continuando se procedió al mapeo de los atributos entre el modelo relacional y el modelo ontológico común. En una etapa final se procedió a la generación del RDF sobre el modelo ontológico común, utilizando los diferentes metadatos extraídos.

En la figura 7.1, se especifican los procedimientos realizados para la generación del modelo ontológico.

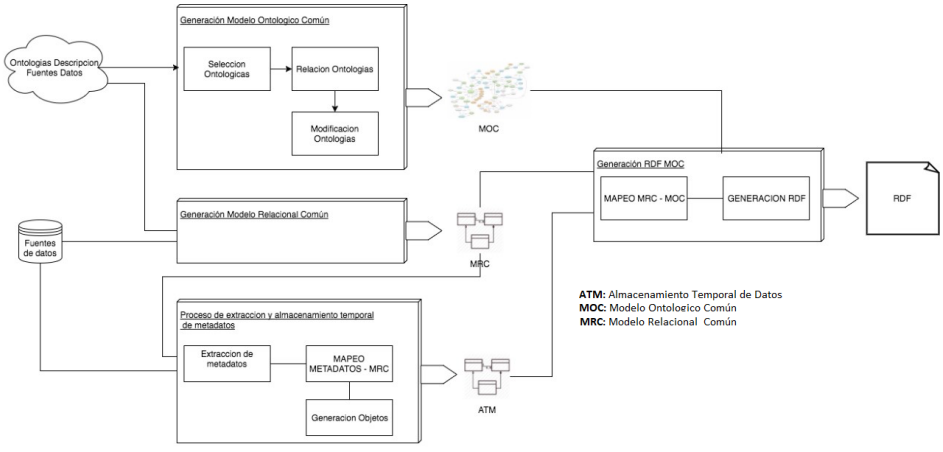


Figura 7.1: Proceso planteado para la generación del modelo ontológico

7.1. Método

Para medir la validez del modelo ontológico creado, se definirá las siguientes fases, especificando en qué procesos del sistema de integración de datos es de utilidad el modelo. En primer lugar se plantea un escenario de integración con diferentes fuentes de datos provistas por el departamento de la Universidad de Cuenca PROMAS. En una segunda fase se especificará las propiedades físicas de las fuentes,así como los datos que almacenan las mismas. Finalmente se definirá el escenario al que se desea llegar con el proceso de integración y la forma en la que el modelo ontológico ayuda a que dicho proceso sea eficiente y exitoso.

Como se define en capítulos anteriores, la finalidad del modelo ontológico común es ayudar al usuario encargado del proceso de integración. En la Figura 7.2 se observa la arquitectura del sistema integrador, donde el esquema mediador es el componente que se relaciona directamente con el usuario. Por lo que la utilidad del modelo ontológico se deberá medir en este componente.

El esquema mediador, es el principal componente en un sistema de integración y así mismo se convierte en el más complejo de obtener. En un caso hipotético de un escenario de integración real se puede tener un gran número de fuentes de datos y la generación del esquema mediador implica la verificación manual de cada una de estas fuentes, convirtiéndose en proceso complejo y costoso.

Para verificar cómo los atributos de las fuente de datos se relaciona con los atributos del esquema mediador se deberá ejecutar una o varias consulta utilizando diferentes herramientas y recursos por cada fuente de datos. Por ejemplo se dispone de 3 fuentes de tipo base de datos, ORACLE, POSTGRES y MYSQL, para determinar qué atributos de estas base de datos corresponden al esquema mediador se deberá lanzar mínimo 3 consultas una por cada fuente. Este problema se resolvería si se tiene una estructura común que almacene todas las características de las 3 fuente de datos y se pueda consultar directamente sobre la misma(Modelo Ontológico Común).

Con lo dicho anteriormente, se puede concluir que el modelo ontológico común, ayudará directamente a la generación del esquema mediador, facilitando al usuario encargado de la integración, la definición de las asignaciones semánticas entre las fuentes de datos y el esquema mediador al que se quiere llegar. Para esto el modelo ontológico representa una estructura común que almacenará las características de las n fuentes de datos que se quieran analizar, estructura que resultará mucho más eficiente y fácil de consultar con el fin de obtener estas asignaciones. Figura 7.2

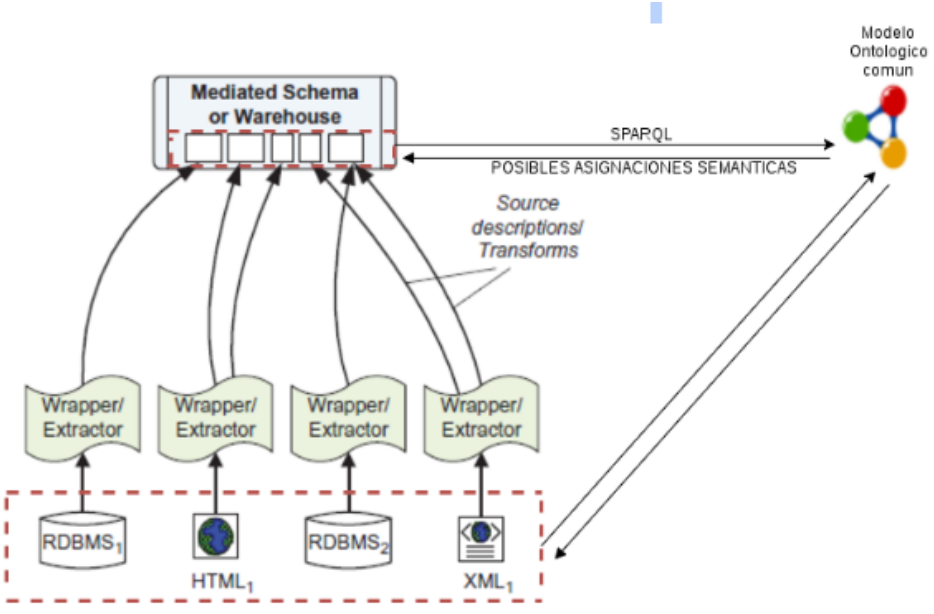


Figura 7.2: Arquitectura básica de un sistema de integración de datos

7.1.1. Ejemplo de un escenario de integración de datos

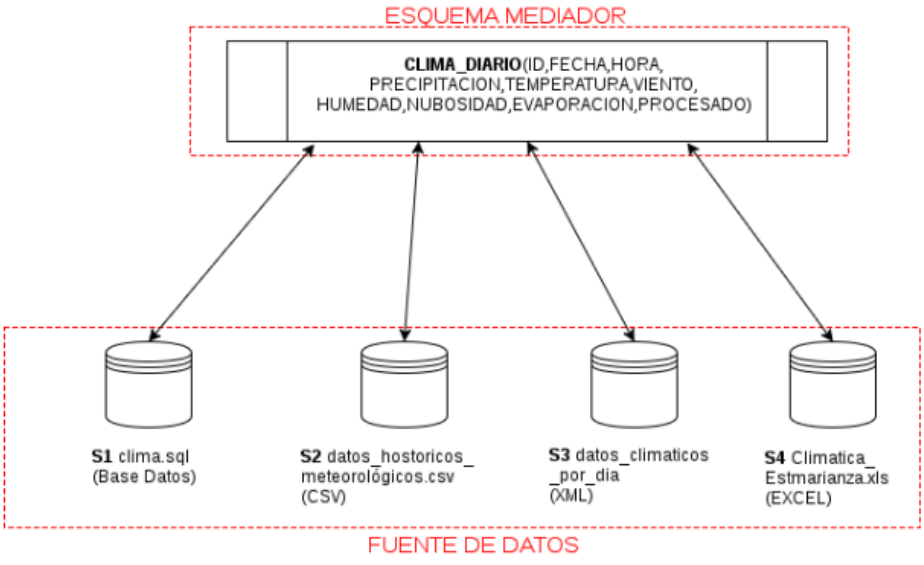


Figura 7.3: Escenario de integración de datos

El siguiente ejemplo, que se muestra en la Figura 7.3, ilustra una integración de datos completa sobre las fuentes de datos provistas.

Fuentes de datos

En el escenario de ejemplo se observa 4 fuentes de datos las mismas que fueron provistas por el departamento PROMAS. Las diferentes fuentes almacenan información referente a muestras recogidas sobre el clima en un periodo de tiempo dado.

La primera fuente a la izquierda denominada como S1, es un conjunto de datos de tipo base de datos, específicamente MYSQL, cuyo modelo de datos está formado por una tabla denominada 'clima_diario' con 11 columnas.

La fuente denominada como S2, es un tipo de archivo CSV, la misma que consta de 21 columnas.

La fuente S3 es un archivo tipo XML el mismo que cuenta con 100 registros

cada uno con 12 propiedades . describe información climática por dia y hora dependiendo cuando se recogió dicha muestra.

Finalmente la fuente S4 es un archivo tipo EXCEL el mismo que consta de 1 Worsheet con 31 columnas.

Fuente	**S1**	**S2**	**S3**	**S4**
format	Mysql	csv	xml	xls
license	GPLv2	GPLv2	GPLv2	GPLv2
keyword	Hydrology, location, temperature, mysql	Hydrology, csv, historico	Climatologia, clima, Meteorología	Meteorología y climatología, Clima, Ciencia
language	es	es	es	es
description	Esta cobertura presenta las estaciones climatológicas del país obtenidas por PROMAS	Esta cobertura presenta las estaciones climatológicas del país obtenidas por PROMAS	Esta cobertura presenta las estaciones climatológicas del país obtenidas por PROMAS	Esta cobertura presenta las estaciones climatológicas del país obtenidas por PROMAS
theme	Datos climáticos	Datos históricos	Temperatura por dia	
modified	2015-12-31	2014-10-09	2013-12-07	1998-12-09

Cuadro 7.1: Describe las propiedades físicas de las fuentes de datos analizadas..

En la Tabla 7.1, se especifica más a detalle la estructura física de las fuentes. El modelo de datos que almacena cada una de ellas se especifica en el ANEXO I.

Cabe recalcar, que las fuentes de datos descrita si bien semánticamente representan el mismo dominio, las mismas son totalmente diferentes sintácticamente por ejemplo cada una posee nombre de atributos diferentes. Así mismo cada fuente almacena su propio modelo de datos lo que significa que el acceso a este para la extracción de los datos sea diferente por cada fuente. Para extraer los datos y metadatos de una base de datos se debe manejar consultas SQL mientras que para un archivo XML se deberá implementar consultas Xquery, por lo que la determinación de las asignaciones semánticas de las fuentes de datos con el esquema mediador resulta tan complejo y costoso dependiendo del número de fuentes que se disponga así como el modelo que sustentan estas.

Esquema Mediador

La definición del esquema mediador es propio del usuario encargado de la integración, el como experto del dominio de la información conoce a qué esquema se

debe llegar. Para el escenario de ejemplo se definió el esquema mediador en base a las fuentes proporcionadas, capturando los atributos más relevantes presentes en estas fuentes.

El esquema mediador será representado como una base de datos con una única relación o tabla denominada clima_diario. Se optó por sólo definir una tabla para explicar de la mejor y más detallada manera la utilidad del modelo ontológico común generado. El propósito de este apartado es demostrar la validez del modelo mas no tratar de simular un proceso de integración cien por ciento real.

En la tabla 7.2 se especifica los atributos o columnas de la tabla clima_diario que representa el esquema mediador deseado.

Nombre Tabla: Clima_diario			
Columna	**Tipo**	**Dominio**	**Descripción**
fecha	Date	YYYY-MM-DD	Representa la fecha el cual se registró los datos.
hora	Date	HH:mm:ss	Representa la hora el cual se registró los datos.
precipitacion	Number(19,3)	Número de decimales:0-3, Valor mínimo: 0mm, Valor máximo: 7mm	La tasa de precipitación, se define como la cantidad de agua líquida o sólida que alcanza el suelo en cierta unidad de tiempo. Los valores de max y min de precipitación se basan en los diferentes modelos de pluviómetros existentes en el mercado q puede abarcar hasta los 500mm/dia. Para la definición del esquema mediador se tomarán valores entre 0 y 7mm
temperatura	Number(19,2)	Número de decimales:0-2, Valor minimo: 0 grados centígrados, Valor máximo: 35 grados centígrados	Rango de valores, basados en las mediciones temperatura normales registrados en el territorio ecuatoriano (0 a 35 grados centígrados). El dominio dependerá del encargado de la integración por ejemplo, si las muestras sólo corresponden a Cuenca, el rango seria (4 a 29 grados)
viento	Number(19,2)	Kilómetros por hora, Número de decimales:0-2, Valor mínimo: 0 Valor máximo: 20	La escala de Beufort es una medida empírica para la intesidad del viento. Según esta escala existen diferentes denominaciones para la velocidad del viento, para este trabajo se tomó en valores considerados desde calma (0 a 1 Kilómetros por hora) hasta Flojo (12-20 Kilómetros por hora). No se tomó en cuenta valores mayores a estos como huracanes o vientos fuertes ya que difícilmente se den estas mediciones[43]

Cuadro 7.2: Estructura del esquema mediador. (1/2)

Nombre Tabla: Clima_diario			
Columna	**Tipo**	**Dominio**	**Descripción**
humedad	Number(19,2)	Número de decimales:0-2, Valor mínimo: 0 por ciento, Valor máximo: 100 por ciento	La humedad indica la cantidad de vapor de agua que se encuentra presente en el aire. La humedad relativa por lo cual solo se puede considerar valores entre 0 mínimo y 100 por ciento
nubosidad	Integer	Valor mínimo: 0, Valor máximo: 8	En meteorología, un okta es una medida utilizada para describir la nubosidad. Rango de 0 a 8, es estimado en términos de cuántos octavos de cielo están cubiertos por las nubes.
procesado	Boolean	Valores: T/F, 0/1, True/False	Representa si el registro ya fue procesado de alguna manera en algún momento

Cuadro 7.3: Estructura del esquema mediador. (2/2)

Una vez definido el esquema mediador de la arquitectura del sistema de integración, lo siguiente es definir las asignaciones semánticas entre este esquema y las fuentes de datos, estos se pueden definir mediante una inspección manual consultado por separado cada fuente de datos sin embargo el propósito de este trabajo es ayudar al usuario en el proceso de asignación, por lo que en el siguiente apartado se especificará las asignaciones semánticas obtenidas tanto de manera manual así como mediante la utilización del modelo ontológico común.

Construcción del modelo mediador

Como primer paso para la definición de las asignaciones semánticas se genero el RDF sobre el modelo ontológico común utilizando los metadatos de las diferentes fuentes de datos.

Para la generación de RDF se utilizó una aplicación JAVA con tecnología maven la cual recibe como entrada los parámetros necesarios para la conexión y extracción de los metadatos de cada fuente de datos. En la tabla 7.4 se especifica los parámetros de entrada a la aplicación por cada fuente de datos.

La aplicación adicionalmente recibe como entrada el porcentaje de los datos que se desea analizar para determinar la estructura(*StorageFormat*). Finalmente el modelo ontológico común se puede considerar como una entrada mas a la aplicación sobre la cual se generará el RDF que sirve como repositorio común de todos los metadatos de cada una de las fuentes.

Fuente de datos	Tipo	Parámetros entrada
clima.sql	DB	hostname, puerto, ip, esquema
datos_hostoricos_meteorológicos.csv	CSV	Path de acceso a la fuente.
Climatica_Estmarianza.xls	EXCEL	Path de acceso a la fuente.
datos_climaticos_por_dia.xml	XML	Path de acceso a la fuente.

Cuadro 7.4: Parámetros de entrada para las diferentes fuentes

En la figura 7.4 se resume el proceso de generación de RDF, a través de la aplicación creada especificando las diferentes herramientas utilizadas.

Generación de asignaciones semánticas

Como se explicó anteriormente,en la definición de las asignaciones semánticas entre las fuentes de datos y el esquema mediador se pueden utilizar diferentes lenguajes de mapeo de schema. Entre los más conocidos se tiene: Global as View (GAV), Local as View (LAV) y GLAV.

En el ANEXO G, se especifican las diferentes consultas que se realizaron sobre el modelo ontológico común con la finalidad de obtener posibles asignaciones semánticas. En la tabla se especifica el atributo de esquema mediador, las consultas realizadas y los datos resultantes como posibles asignaciones semánticas. Para definir los atributos que resultaron como posibles asignaciones semánticas, se especifica el dataset, la tabla y la columna del que proviene por ejemplo: *Dataset:Table:Columna*.

En la tabla 7.5, se especifica en cambio las asignación semánticas encontrados mediante una inspección manual y mediante el modelo ontológico común, especificando cuántas columnas se tuvo que analizar en cada caso y la validez de cada método.

7.1.2. Interpretación de resultados y validez del modelo ontológico

En el Anexo G, Se especifica las diferentes asignaciones semánticas obtenidas para cada atributo del esquema mediador, a través de varias consultas realizadas sobre el modelo ontológico común.

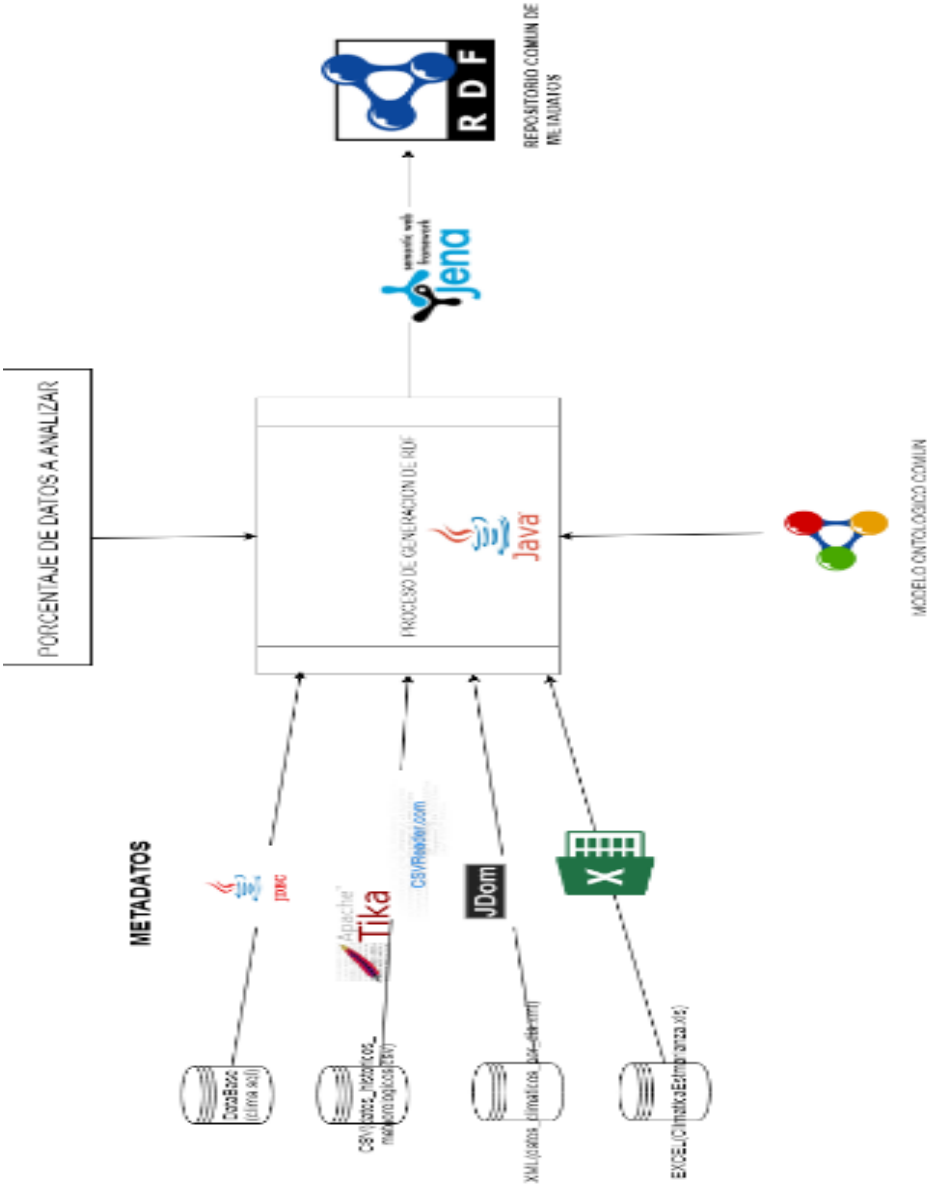


Figura 7.4: Proceso de generación del RDF

En la tabla 7.5 se indica las asignaciones semánticas correctas, obtenidas mediante una inspección manual sobre las fuentes de datos. Adicionalmente, se muestra las asignaciones obtenidas mediante el modelo ontológico común, así como el número de asignaciones correctas que arrojó la consulta sobre el modelo ontológico.
El número de asignaciones correctas, se considera cuántos valores correctos devuelve la consulta sobre el modelo ontológico, más no, el valor total de valores devueltos. Los valores incorrectos devueltos de igual manera se especifica en la tabla 7.5. La nomeclatura que se presenta en las tablas es la siguiente: T = Total de asignaciones correctas. AC = Asignaciones correctas recuperadas. AI = Asignaciones incorrectas recuperadas

Como se observa en la Tabla 7.5, por cada atributo del esquema mediador se realizan varios tipos de consulta en las cuales se devuelven el mismo valor. Por lo que se hará en conceptos matemáticos una unión de todos los valores devueltos de todas las consultas realizadas por cada atributo para especificar el número de asignación semánticas a través del modelo ontológico.

Atributo Esquema Mediador	**Asignaciones Semánticas Correctas (Inspección Manual)**	**Asignaciones Semánticas (Modelo Ontológico)**	**T**	**AC**	**AI**
fecha	▪ Climatica_Esta marianza.xls:Clima tica_Estamarianza05082015:Date ▪ datos_climaticos_por_dia.xml/mes:dia:Dia ▪ datos_historicos_meteorologicos.csv:datos_hist oricos_meteorologicos:Month ▪ datos_historicos_meteorologicos.csv:datos_hist oricos_meteorologicos:Day ▪ datos_historicos_meteor ologicos.csv:datos_hist oricos_meteorologicos:Month ▪ datos_historicos s_meteo rologicos.csv:datos_hist oricos_meteorologicos:Year	▪ datos_climaticos_por_dia.xml/mes:dia:Dia ▪ ClimaticaEsta marianza.xls:Climatica_Estamarianza05082015:Date ▪ Climatica_Estam rianza.xls:Climatica_Estamarianza05082015:Time ▪ datos_climaticos_por_dia.xml/mes:dia:hora ▪ datos_historicos_meteorologicos.csv:datos_hist oricos_meteorologicos:Month ▪ datos_historicos_meteorologicos.csv:datos_hist oricos_meteorologicos:Day ▪ datos_historicos_meteorologicos.csv:datos_hist oricos_meteorologicos:Year	5	5	2

hora	▪ Climatica_Esta marianza.xls:Climatica_Estamarianza05082015:Time ▪ datos_climaticos_por_dia.xml/mes:hora:Hora ▪ datos_historicos_meteorologicos.csv:datos_hist oricos_meteorologicos:Minute ▪ datos_historicos_meteorologicos.csv:datos_hist oricos_meteorologicos:Hour	▪ ClimaticaEsta marianza.xls:Climatica_Estamarianza05082015:Time ▪ datos_climaticos_por_dia.xml/mes:hora:Hora ▪ Climatica_Esta marianza.xls:Climatica_Estamarianza05082015:Date ▪ datos_climaticos_por_dia.xml/mes:dia:Dia ▪ datos_historicos_meteorologicos.csv:datos_hist oricos_meteorologicos:Month ▪ datos_historicos_meteorologicos.csv:datos_hist oricos_meteorologicos:Minute ▪ datos_historicos_meteorologicos.csv:datos_hist oricos_meteorologicos:Cloudy ▪ datos_historicos_meteorologicos.csv:datos_hist oricos_meteorologicos:Hour ▪ datos_historicos_meteorologicos.csv:datos_hist oricos_meteorologicos:Historical Record ▪ datos_historicos_meteorologicos.csv:datos_hist oricos_meteorologicos:Day ▪ datos_historicos_meteorologicos.csv:datos_hist oricos_meteorologicos:Shushine time[sfc]	4	4	7

precipitacion	▪ Climatica_Estamarianza.xls:Climatica_Estamarianza05082015:In Precipitation [mm] ▪ Climatica_Estamarianza.xls:Climatica_Estamarianza05082015:In Precipitation [cm] ▪ datos_historicos_meteorologicos.csv:datos_historicos_meteorologicos:Total Precipitation [cm] ▪ datos_climaticos_por_dia.xml/mes:Meteoros:Precip..._a_50mm	▪ datos_climaticos_por_dia.xml/mes:Meteoros:Precip..._a_50mm ▪ datos_climaticos_por_dia.xml/mes:Meteoros:cloud1..._a_0cm ▪ datos_climaticos_por_dia.xml/mes:Meteoros:Nivel2..._a_0cm ▪ datos_climaticos_por_dia.xml/mes:Met eoros:Nivel1..._a_0cm ▪ Climatica_Estamarianza.xls:Climatica_Estamarianza05082015:Processed Record ▪ Climatica_Estamarianza..xls:Climatica_Estamarianza05082015:Wind Speed ▪ Climatica_Estamarianza..xls:Climatica_Estamarianza05082015:In Precipitation [mm] ▪ datos_historicos_meteorologicos.csv:datos_historicos_meteorologicos:Total Precipitation [cm] ▪ Climatica_Estamarianza..xls:Climatica_Estamarianza05082015:In Precipitation [cm]	4	4	5

Cuadro 7.5: Asignaciones Semánticas obtenidas de manera manual y mediante el modelo ontológico común.

En el anexo H, se especifica todas las asignaciones semánticas obtenidas entre las diferentes columnas del esquema mediador y las fuentes de datos de ejemplo.

Sobre las posibles asignaciones semánticas obtenidas mediante el modelo ontológico, se pueden hacer las siguientes consideraciones.

1. En la mayoría de los casos, para cada atributo del esquema mediador se obtiene el 100 % de las asignaciones semánticas correctas. Esto es claro ya

que el usuario encargado de la integración conoce a fondo el dominio del esquema mediador al que se quiere llegar. Por ejemplo para el caso de la precipitación, se define un escenario específico basado en las precipitaciones anuales en el Ecuador lugar donde se recogen las muestras. Es así que se tiene un alto grado de certeza de que ninguna muestra que represente la precipitación tendrá un valor fuera del rango de 0 mm a 7 mm.

Adicionalmente por cada atributo se lanza varias consultas con el fin de contemplar todas las posibles asignaciones semánticas,para el mismo caso de la precipitación se lanza consultas para obtener tanto en milímetros que es el dominio del esquema mediador, pero adicionalmente se consulta en centímetros y metros asumiendo que puede existir algún dataset que almacene la información en estos tipos de medida, diferentes, pero que significan semánticamente lo mismo.

Si bien realizar varias consultas favorece a contemplar todos los posibles asignaciones semánticas, esto trae consigo recuperar valores incorrectos que no representan semánticamente lo mismo, pero que están en el rango de la consulta. Un ejemplo claro de esto, es la temperatura. En este atributo, se lanzó dos consultas, tanto para obtener posibles asignaciones semánticas en grados centígrados como en grados fereigeth, lo que conllevo a obtener el 100 % de las asignaciones correctas. sin embargo esto trajo consigo varios atributos incorrectos o inválidas ya que el rango que se consulta era muy general. Los valores decimales entre 0 y 95 para el caso de la temperatura en grados fareigeth trajo consigo varios valores incorrectos, pero sin perder las correctas asignaciones semánticas y si bien son atributos incorrectos que el encargado de la integración debe descartar manualmente, no es lo mismo evaluar 200 columnas que contienen en total las diferentes fuentes de datos mediante diferentes herramientas de consulta y extracción que simplemente descartar de un grupo de 20 columnas donde 7 son las correctas.

2. Si bien la mayoría de las asignaciones semánticas recuperadas son correctas, existen casos como del atributo Humedad donde se recuperó 2 de las 3 asignaciones correctas. Pero, ¿A se debe esto?, después de analizar manualmente los datos se concluye que el atributo datos_hostoricos_meteorologicos.csv:datos_hostoricos_met humidity [2 m above gnd, no era de tipo Decimal, valor por el cual se realizó la consulta. Este atributo almacena los datos en tipo Entero, pero de igual manera semánticamente hace referencia a la humedad. Sin embargo esto no se trata de un problema del modelo ontológico propiamente, sino más bien del encargado de la integración por no contemplar este caso. Es decir una consulta que recupere los atributos tipo Entero entre 0 y 100 hubiese sido suficiente para recuperar dicho atributo. En otras palabras el modelo ontológico ofrece un sin número de posibilidades para poder consultar sobre

el mismo y quedará ya ha criterio del encargado de la integración el grado de detalle de recuperación de datos al que se quiere llegar.

3. En otro caso específico, de las asignaciones semánticas para el atributo Viento, solo se pudo recuperar 5 de las asignaciones correctas. Así mismo se realizó una inspección manual sobre las fuentes de datos y se pudo concluir que el atributo datos_hostoricos_meteorologicos.csv:datos _hostoricos_meteorologicos:Wind Run almacena valores superiores a 20 km/h, que es el valor máximo que se planteó para este atributo en la definición del esquema mediador. Esto se trata de otro problema más del encargado de la integración que del modelo ontológico en sí, ya que este desconocimiento del dominio del problema traería estos errores. O la introducción de ruido o la obtención incorrecta de la muestras de datos en algún Dataset de igual manera contribuyen a este problema.

4. Otro problema que se puede presentar, aunque en este proceso estuvo ausente, es que el atributo que corresponde a una asignación semántica correcto,a de algún campo del esquema mediado. Contenga un porcentaje menor al que se toma como referencia en la consulta. Es decir en este caso de ejemplo se consideró para todos los atributos del esquema medidor un porcentaje de similitud del 70 por ciento. Sin embargo puede ser que existan atributos que semánticamente representen la información buscada es decir sean asignaciones semánticas correctas pero en un porcentaje menor. Esto debido a tiene una heterogeneidad en sus valores, por diversos factores como ruido, precisión en la obtención de la muestra etc. Por ejemplo puede existir atributos en los cuales el 50 % sea de tipo decimal, y el otro 50 % sea de tipo entero. Si se consulta sobre el 70 %, este atributo no será recuperado aunque semánticamente represente lo que se está buscando. Esto igual se soluciona con el nivel de detalle de consulta y extracción que tenga el desarrollado sobre el modelo ontológico.

5. Finalmente se puede observar que atributos como Procesado o Nubosidad tienen un 100 % de asignaciones semánticas correctas y así mismo un número muy bajo de atributos incorrectos recuperados. Esto se debe a que el rango de búsqueda de estos atributos está definido mucho más a detalle. Por ejemplo en el caso del atributo Procesado al consultar los atributos de tipo Boolean, en el caso de existir será muy pocos lo que cumplan esta condición. De igual manera si se consulta los atributos de tipo Entero con solo dos posibles valores 0 y 1.

Capítulo 8

Conclusiones y trabajos futuros

8.0.3. Conclusiones

- Una de las tareas más costosas y complejas dentro del proceso de integración es la generación de asignaciones semánticas entre el esquema mediador y las fuentes de datos. por cada fuente de datos se debe realizar diferentes consultas sobre los datos mediante diferentes herramientas y recursos por lo cual el costo y la complejidad crece exponencialmente, dependiendo tanto del modelo de datos de la fuente así como de la cantidad de las mismas.
- Si bien una fuente de datos se puede describir a través de sus metadatos, es importante indagar en los datos de la misma a fin de recuperar información importante de la fuente y descubrir otras peculiaridades como presencia de ruido o inconsistencia.
- La unión de las ontologías seleccionadas en este caso PHDD, DCAT y DISCO nos permiten describir la fuente de datos mediante sus metadatos y mediante la estructura denominada storage format describimos la fuente más a detalle analizando los datos de la misma en un porcentaje dado por el usuario encargado de la integración.
- El usuario encargado de la integración podrá ejecutar n consultas sobre el modelo ontológico común a fin de obtener todas las asignaciones semánticas correctas. Donde la validez de estas asignaciones dependerá del nivel de detalle que ejecute el desarrollador en sus consultas. Un buen conocimiento del dominio del esquema mediador al que se pretende llegar permitirá determinar con mucha más exactitud las diferentes asignaciones semánticas, ya que permite lanzar consultas mucho más específicas.

- La obtención de asignaciones semánticas correctas depende en gran medida de que tan específica sea la consulta realizada sobre el modelo ontológico común, por ejemplo una consulta de la forma 'obtener todas las columnas en la cual sus datos sean en un 80 % de tipo decimal, su valor máximo sea x, su valor mínimo sea y, su promedio está entre a y b, su número decimales este entre c y d, su separador decimal default sea el punto (.)' arrojará unas posibles asignaciones semánticas mucho más válidas que al ejecutar una consulta tipo obtener todas las columnas en la cual sus datos sean en un 80 %de tipo decimal, es decir, el modelo ontológico común ,provee al usuario un repositorio general en el cual a través de metadatos y datos describen las fuentes que se vayan a analizar y dependerá del usuario el número de consultas y el nivel de detalle de las mismas que ejecute para obtener las asignaciones semánticas correctas.

- El objetivo general de modelo ontológico común es proporcionar al usuario encargado de la integración un repositorio común o único donde se puedan consultar y extraer información para determinar posibles asignaciones semánticas. sin embargo el modelo ayuda también a la identificación de ruido presente en las fuentes de datos ya que brinda información sobre los diferentes tipo de datos que tienen las diferentes columnas permitiendo al usuario ver qué columnas presentan inconsistencias de información. por ejemplo una columna donde el 80 % de los datos son de tipo decimal y el 20 % de tipo string significa que esta columna requiere especial atención.

- La presencia de ruido o la inconsistencia de datos puede repercutir en la obtención de las asignaciones semánticas correctas. por ejemplo una columna puede ser una asignación semántica correcta de un atributo del esquema mediador sin embargo en un porcentaje menor requerido por el usuario. Un porcentaje menor debido a que el resto de los datos podrían ser otro tipo de formato, datos nulos, presencia de ruido entre otras. Aquí otra vez depende del usuario y la contemplación de todos los casos para lanzar todas las consultas necesarias, en este caso ir probando con diferentes porcentajes de validez.

8.0.4. Trabajos Futuros

- En este trabajo se considero únicamente fuente de datos de tipo SQL, CSV, XML y EXCEL. Sin embargo el modelo es completamente extensible a otras fuentes como JSON, Web Services, Archivos de texto plano, Documentos HTML, entre otros. Los cuales se pueden migrar a un modelo relacional formado por tablas y columnas.

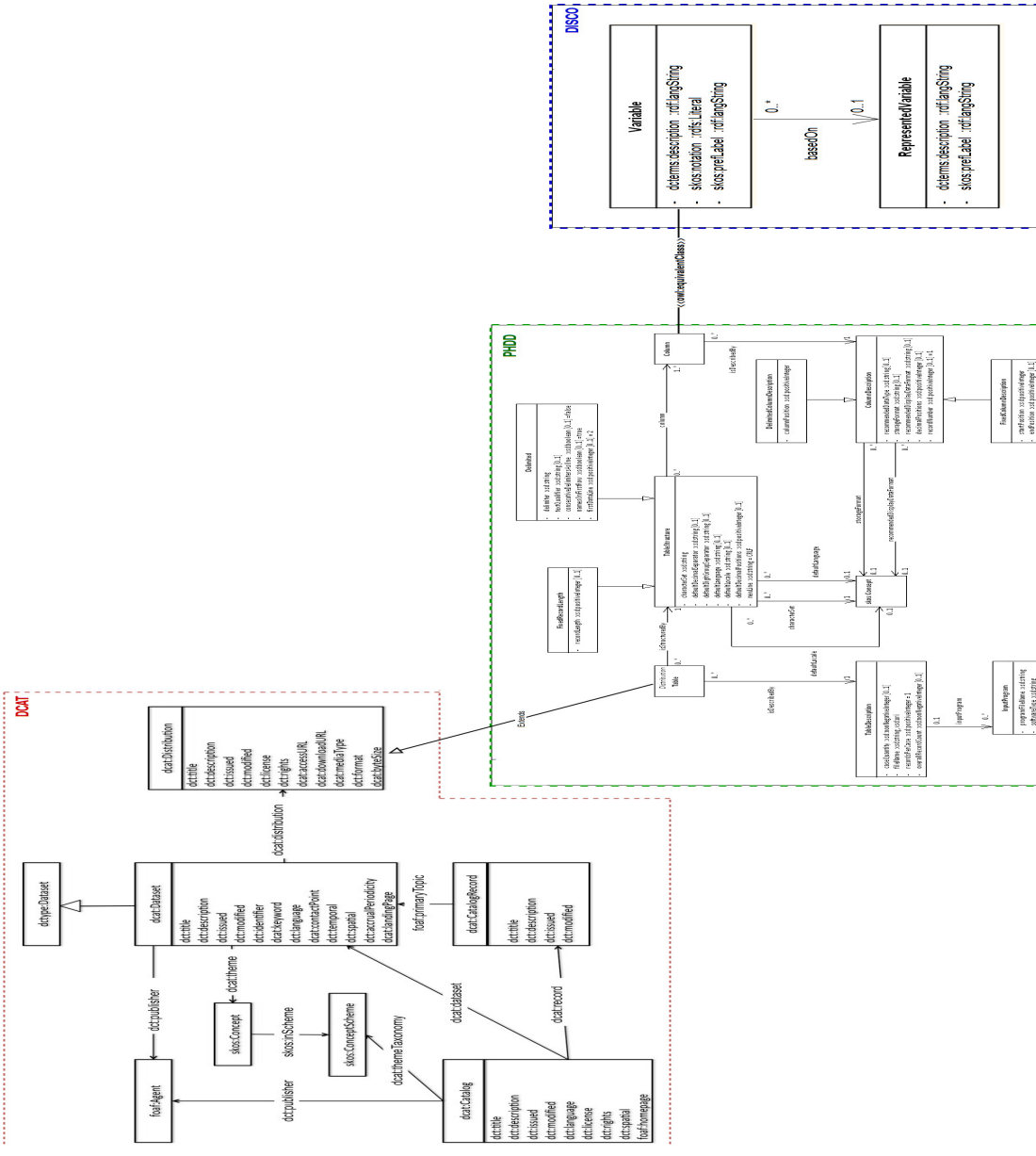
- Para la obtención de la estructura StorageFormat , se tuvo que analizar los datos de cada fuente de datos en un porcentaje proporcionado por el usuario.

Se pudo observar que el proceso de determinación del tipo de dato , resulta usado para ser ejecutado de manera lineal, por lo que este proceso se podría llevar a cabo en un futuro mediante clusterización o mediante base de datos NoSQL. A fin de lograr un mayor rendimiento.

- Sobre el modelo ontológico generado ha futuro se podría implementar técnicas de String matching. Con la finalidad de fortalecer más el mismo .

ANEXO A. Modelo ontológico general completo, resultado de la unión de los 3 vocabularios

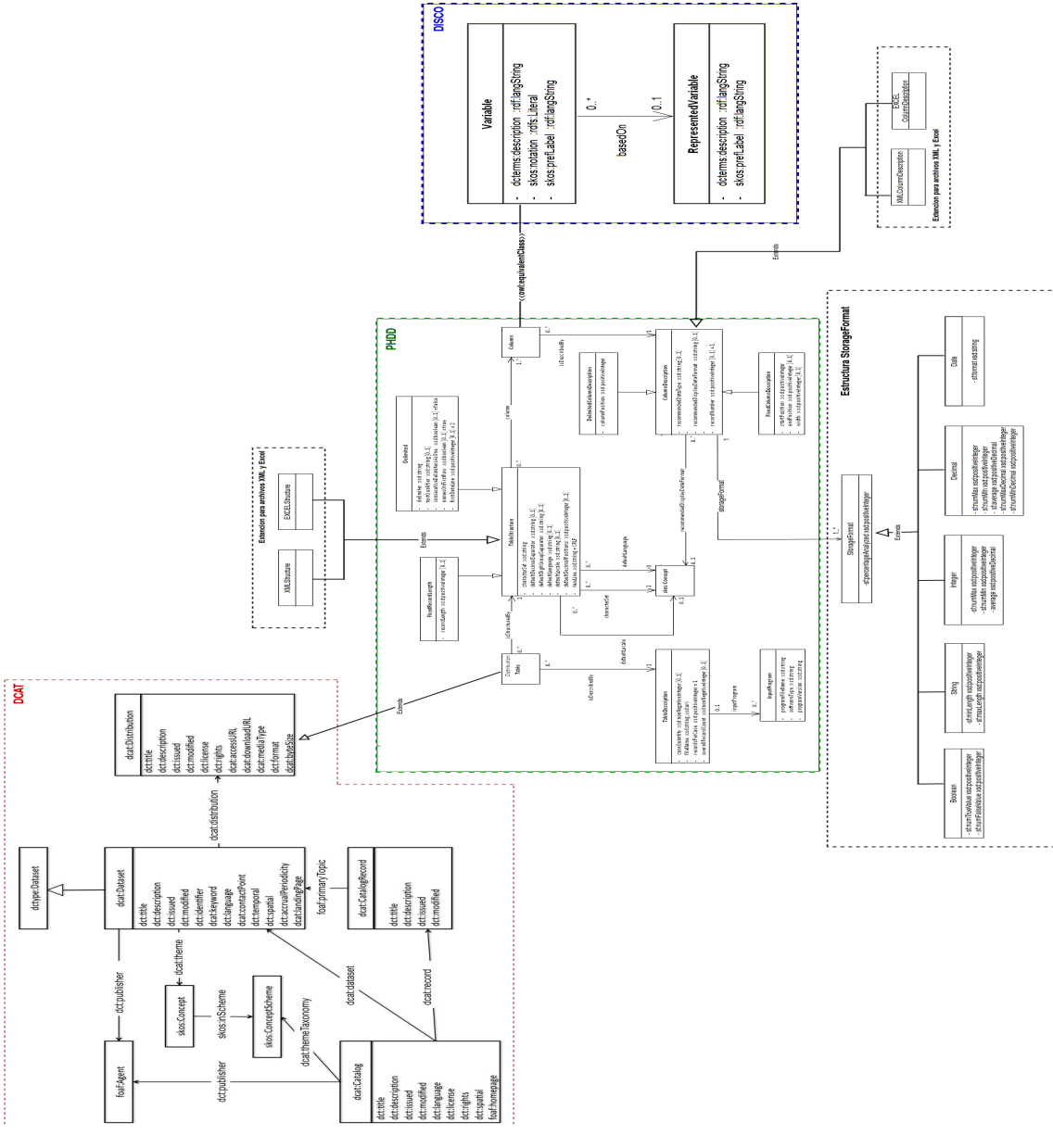
EL ANEXO A, se adjunta de manera digital bajo el nombre ANEXOA.jpeg, para una mejor visualización del archivo, debido a que se trata de una imagen bastante grande.



ANEXO B. Modelo ontológico general completo

EL ANEXO B, se adjunta de manera digital bajo el nombre ANEXOB.png, para una mejor visualización del archivo, debido a que se trata de una imagen

bastante grande.



ANEXO C. Mapeo de los metadatos extraídos del archivo CSV con el modelo relacional común. Clases TABLE_CSV y COLUMN_CSV

Atributo Modelo Relacional	**Herramienta**	**Proceso/Método**
title	java.io.File	getName()
description		Extracción manual
modified	java.nio.file	lastModifiedTime()
issued	java.nio.files	creationTime()
license		Extracción manual
rights		Extracción manual
downloadURL		Extracción manual
license		Extracción manual
mediaType		application/csv
byteSize	java.io.File	length()
format		application/csv
characterSet	org.apache.tika	metadata.get(Çontent-Type")

Cuadro 8.1: Extracción metadatos y mapeo con el modelo relacional (Clase TABLE_CSV) (1/2)

Atributo Modelo Relacional	**Herramienta**	**Proceso/Método**
defaultDecimalSeparator		Proceso programado mediante análisis de datos
defaultDigitGroupSeparator		Proceso programado mediante análisis de datos
defaultLanguage	org.apache.tika	metadata.get(Çontent-Language")
defaultDecimalPositions		Proceso programado mediante análisis de datos
newLine		Extracción manual
recordPerCase		Extracción manual
overrallRecordCount	org.apache.tika	readRecord()
programFileName		Proceso programado mediante análisis de datos
softwareType		Proceso programado mediante análisis de datos
programVersion		Proceso programado mediante análisis de datos
delimiter	com.csvreader.CsvReader	getDelimiter()
textQualifier	com.csvreader.CsvReader	getTextQualifier()
firstDataLine		Proceso programado mediante análisis de datos
consecutiveDelimiterAsOne		Proceso programado mediante análisis de datos
nameOnFirstRow		Proceso programado mediante análisis de datos

Cuadro 8.2: Extracción metadatos y mapeo con el modelo relacional (Clase TABLE_CSV) (2/2)

Atributo Modelo Relacional	Herramienta	Proceso/Método
prefLabel	com.csvreader.CsvReader	Proceso programado mediante análisis de datos
description		Extracción manual
recordNumber	com.csvreader.CsvReader	Proceso programado mediante análisis de datos
columnPosition	com.csvreader.CsvReader	Proceso programado mediante análisis de datos

Cuadro 8.3: Extracción metadatos y mapeo con el modelo relacional (Clase COLUMN_CSV)

ANEXO D. Extracción de metadatos para la fuente de datos de ejemplo (Base de Datos) y mapeo con el modelo relacional

Atributo Modelo Relacional	Proceso	Resultado
prefLabel	SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME =	fecha
record_number	SELECT count('COLUMN') FROM SCHEMA.TABLE WHERE 'COLUMN' IS NOT NULL	38420
description	SELECT COLUMN_COMMENT FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME = and COLUMN_NAME ='	

Cuadro 8.4: Ejemplo de extracción de metadatos y mapeo con el modelo relacional(Columna fecha)

Atributo Modelo Relacional	**Proceso**	**Resultado**
prefLabel	SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME =	precipitacion
record_number	SELECT count('COLUMN') FROM SCHEMA.TABLE WHERE 'COLUMN' IS NOT NULL	32500
description	SELECT COLUMN_COMMENT FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME = and COLUMN_NAME ='	

Cuadro 8.5: Ejemplo de extracción de metadatos y mapeo con el modelo relacional(Columna precipitacion)

Atributo Modelo Relacional	**Proceso**	**Resultado**
prefLabel	SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME =	nubosidad
record_number	SELECT count('COLUMN') FROM SCHEMA.TABLE WHERE 'COLUMN' IS NOT NULL	38420
description	SELECT COLUMN_COMMENT FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME = and COLUMN_NAME ='	

Cuadro 8.6: Ejemplo de extracción de metadatos y mapeo con el modelo relacional(Columna nubosidad)

Atributo Modelo Relacional	**Proceso**	**Resultado**
prefLabel	SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME =	evaporacion
record_number	SELECT count('COLUMN') FROM SCHEMA.TABLE WHERE 'COLUMN' IS NOT NULL	28963
description	SELECT COLUMN_COMMENT FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME = and COLUMN_NAME ='	

Cuadro 8.7: Ejemplo de extracción de atributos/metadato y mapeo con el modelo relacional(Columna evaporacion)

Atributo Modelo Relacional	**Proceso**	**Resultado**
prefLabel	SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME =	temperatura
record_number	SELECT count('COLUMN') FROM SCHEMA.TABLE WHERE 'COLUMN' IS NOT NULL	28563
description	SELECT COLUMN_COMMENT FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME = and COLUMN_NAME ='	

Cuadro 8.8: Ejemplo de extracción de metadatos y mapeo con el modelo relacional(Columna temperatura)

Atributo Modelo Relacional	Proceso	Resultado
prefLabel	SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME =	viento
record_number	SELECT count('COLUMN') FROM SCHEMA.TABLE WHERE 'COLUMN' IS NOT NULL	35000
description	SELECT COLUMN_COMMENT FROM INFORMATION_SCHEMA.COLUMNS WHERE SCHEMA_NAME = and TABLE_NAME = and COLUMN_NAME ='	

Cuadro 8.9: Ejemplo de extracción de metadatos y mapeo con el modelo relacional(Columna viento)

ANEXO E. Mapeo completo entre el modelo ontológico común y el modelo relacional general

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
DATASET		dcat:Dataset	Class
DATASET	title	dct:title	Property
DATASET	description	dct:description	Property
DATASET	issued	dct:issued	Property
DATASET	modified	dct:modified	Property
DATASET	language	dct:language	Property
DATASET	publisher	dct:publisher	Property
DATASET	accrualPeriodicity	dct:accrualPeriodicity	Property
DATASET	keyword	dcat:keyword	Property
DATASET	identifier	dct:identifier	Property
DATASET	percentageAnalized	sf:percentageAnalized	Property

Cuadro 8.10: Mapeo propiedades DATASET - dcat:Dataset

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
TABLE		dcat:Distribution/ phdd:Table	Class
TABLE	title	dct:title	Property
TABLE	description	dct:description	Property
TABLE	issued	dct:issued	Property
TABLE	modified	dct:modified	Property
TABLE	license	dct:license	Property
TABLE	rigths	dct:rigths	Property
TABLE	accessUrl	dcat:accessUrl	Property
TABLE	downloadUrl	dcat:downloadUrl	Property
TABLE	mediaType	dcat:mediaType	Property
TABLE	format	dct:format	Property
TABLE	byteSize	dcat:ByteSize	

Cuadro 8.11: Mapeo propiedades TABLE - dcat:Distribution

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
TABLE_CSV		phdd:Delimited	Class
TABLE_CSV	delimiter	phdd: delimiter	Property
TABLE_CSV	textQualifier	phdd: textQualifier	Property
TABLE_CSV	consecutiveDe limite-rAsOne	phdd:consecutiveDe limiterAsOne	Property
TABLE_CSV	nameOnFirst Row	phdd: nameOnFirst Row	Property
TABLE_CSV	firstDataLine	phdd: firstDataLine	Property

Cuadro 8.12: Mapeo propiedades TABLECSV - phdd:Delimited

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
TABLE		phdd:TableStructure	Class
TABLE	characterSet	phdd:CharacterSet	Property
TABLE	defaultDecimalSeparator	phdd: defaultDecimalSeparator	Property
TABLE	defaultDigitGroupSeparator	phdd: defaultDigitGroupSeparator	Property
TABLE	defaultLanguage	phdd: defaultLanguage	Property
TABLE	defaultLocale	phdd: defaultLanguage	Property
TABLE	defaultDecimalPositions	phdd: defaultDecimalPositions	Property

Cuadro 8.13: Mapeo propiedades TABLE - phdd:TableStructure

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
TABLE		phdd:TableDescription	Class
TABLE	caseQuantity	phdd:caseQuantity	Property
TABLE	recordPerCase	phdd: recordPerCase	Property
TABLE	overallRecordCount	phdd: overallRecordCount	Property

Cuadro 8.14: Mapeo propiedades TABLE - phdd:TableDescription

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
TABLE		phdd:InputProgram	Class
TABLE	programFileName	phdd: programFileName	Property
TABLE	softwareType	phdd: softwareType	Property
TABLE	programVersion	phdd: programVersion	Property

Cuadro 8.15: Mapeo propiedades TABLE - phdd:InputProgram

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
COLUMN		phdd:Column	Class
COLUMN	prefLable	skos:prefLabel	Property
COLUMN	description	dcterms: description	Property

Cuadro 8.16: Mapeo propiedades COLUMN - phdd:Column

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
COLUMN		phdd:Column Description	Class
COLUMN	recordNumber	phdd:recordNumber	Property

Cuadro 8.17: Mapeo propiedades COLUMN - phdd:ColumnDescription

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
COLUMN_CSV		phdd:Delimited ColumnDescription	Class
COLUMN_CSV	columnPosition	phdd:columnPosition	Property

Cuadro 8.18: Mapeo propiedades COLUMN_CSV - phdd:DelimitedColumnDescription

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
STORAGEFORMAT		dduc:StorageFormat	Class
STORAGEFORMAT	Percentage	sf:percentageAnalyzed	Property

Cuadro 8.19: Mapeo propiedades STORAGEFORMAT - sf:StorageFormat

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
STRING		sf:String	Class
STRING	minLength	sf:minLength	Property
STRING	maxLength	sf:maxLength	Property

Cuadro 8.20: Mapeo propiedades STRING - sf:String

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
INTEGER		sf:Integer	Class
INTEGER	numMax	sf:numMax	Property
INTEGER	numMin	sf:numMin	Property
INTEGER	average	sf:average	Property

Cuadro 8.21: Mapeo propiedades INTEGER - sf:Integer

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
DECIMAL		sf:Decimal	Class
DECIMAL	numMax	sf:numMax	Property
DECIMAL	numMin	sf:numMin	Property
DECIMAL	average	sf:average	Property
DECIMAL	numMaxDecimal	sf:numMaxDecimal	Property
DECIMAL	numMinDecimal	sf:numMinDecimal	Property

Cuadro 8.22: Mapeo propiedades DECIMAL - sf:Decimal

Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
DATE		sf:Date	Class
DATE	format	sf:format	Property

Cuadro 8.23: Mapeo propiedades DATE - sf:Date

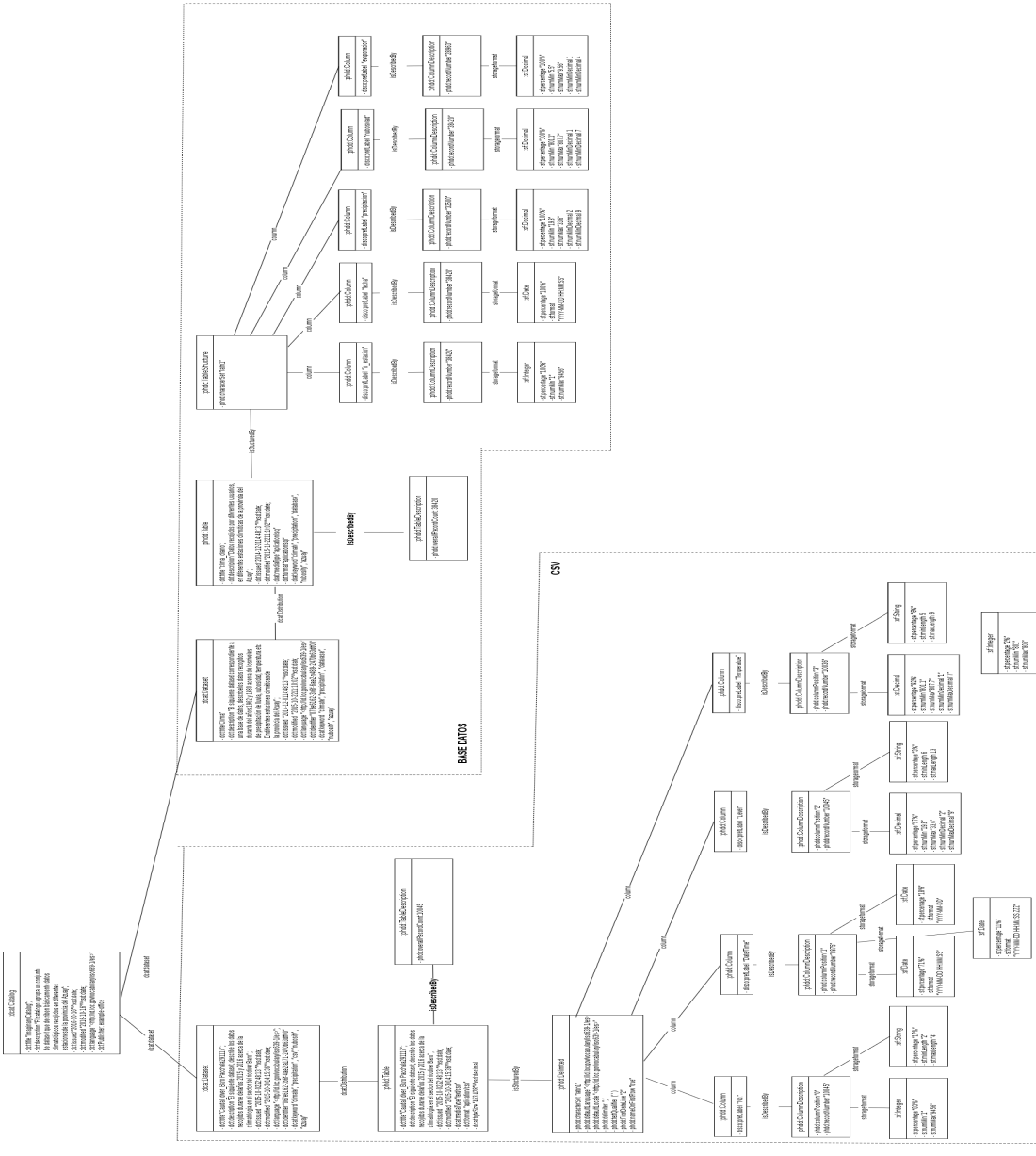
Modelo Relacional común		Modelo ontológico común	
Nombre Clase	**Atributo**	**Propiedad RDF**	**Tipo**
BOOLEAN		sf:Boolean	Class
BOOLEAN	numTrueValue	sf:format	Property
BOOLEAN	numTrueFalse	sf:format	Property

Cuadro 8.24: Mapeo propiedades BOOLEAN - sf:Boolean

ANEXO F. RDF generado sobre el modelo ontológico común para el caso del conjunto de datos CSV y Base de Datos

EL ANEXO F, se adjunta de manera digital bajo el nombre ANEXOF.png, para una mejor visualización del archivo, debido a que se trata de una imagen

bastante grande.



ANEXO G. Determinación de asignaciones semánticas entre el esquema mediador y las fuentes de datos

Atributo Esquema Mediador:	FECHA		
CONSULTAS DE EXTRACCIÓN DE DATOS	**CONSULTA SPARQL**	**ATRIBUTOS RESULTANTES**	**% DATOS ANALIZADOS**
Todos los atributos en los cuales, minimo, el 70% del total de sus valores sean de tipo sf:Date con un formato 'yyyy-mm-dd'	SELECT (?x AS ?Format) (?c AS ?Porcentaje) (?z AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset) (?a AS ?FormatDate) WHERE { ?x rdf:type sf:Date . ?x sf:percentage ?c. ?x sf:formatDate ?a . ?y sf:haveStorageFormat ?x . ?z phdd:isDescribedBy ?y . ?w phdd:column ?z . ?b phdd:isStructuredBy ?w . ?b dcterms:title ?f . ?m dcat:distribution ?b . ?m dcterms:title ?k . FILTER(?c >= 70) . FILTER(?a = "yyyy-mm-dd") .}	• datos_climaticos_por_dia.xml/mes:dia:Dia	**70%**
Todos los atributos en los cuales, minimo, el 70% del total de sus valores sean de tiposf:Date con un formato 'dd/MM/yyyy'	SELECT (?x AS ?Format) (?c AS ?Porcentaje) (?z AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset) (?a AS ?FormatDate) WHERE { ?x rdf:type sf:Date . ?x sf:percentage ?c. ?x sf:formatDate ?a . ?y sf:haveStorageFormat ?x . ?z phdd:isDescribedBy ?y . ?w phdd:column ?z . ?b phdd:isStructuredBy ?w . ?b dcterms:title ?f . ?m dcat:distribution ?b . ?m dcterms:title ?k . FILTER(?c >= 70) . FILTER(?a = "dd/MM/yyyy") .}	• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Date	**70%**
Todos los atributos en los cuales, minimo, el 70% del total de sus valores sean de tipo sf:Date con cualquier tipo de formato	SELECT (?x AS ?Format) (?c AS ?Porcentaje) (?z AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset) (?a AS ?FormatDate) WHERE { ?x rdf:type sf:Date . ?x sf:percentage ?c. ?x sf:formatDate ?a . ?y sf:haveStorageFormat ?x . ?z phdd:isDescribedBy ?y . ?w phdd:column ?z . ?b phdd:isStructuredBy ?w . ?b dcterms:title ?f . ?m dcat:distribution ?b . ?m dcterms:title ?k . FILTER(?c >= 70) .}	• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Date • datos_climaticos_por_dia.xml/mes:dia:DiaClimatica_Estmarianza.xls:Climatica_Estamarianza05082015:Time • datos_climaticos_por_dia.xml/mes:dia:hora	**70%**

Figura 8.1: Consulta sobre el modelo ontológico común: atributo Fecha

Atributo Esquema Mediador:	Fecha		
CONSULTAS DE EXTRACCIÓN DE DATOS	**CONSULTA SPARQL**	**ATRIBUTOS RESULTANTES**	**% DATOS ANALIZADOS**

Todos los atributos en los cuales, minimo, el 70% del total de sus valores sean de tipo Entero, su numero minimo sea mayor a cero y su numero maximo menor igual a 31 (DIA)	SELECT ?nmin ?nmax (?x AS ?Format) (?c AS ?Porcentaje) (?z AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset)
	WHERE {
	?x rdf:type sf:Integer .
	?x sf:percentage ?c.
	?x sf:numeroMinimoInteger ?nmin .
	?x sf:numeroMaximoInteger ?nmax .
	?y sf:haveStorageFormat ?x .
	?z phdd:isDescribedBy ?y .
	?w phdd:column ?z .
	?b phdd:isStructuredBy ?w .
	?b dcterms:title ?f .
	?m dcat:distribution ?b .
	?m dcterms:title ?k .
	FILTER(?c >= 70) .
	FILTER(?nmin >= 0 && ?nmax <= 31) .
	}
		datos_hostoricos_meteorologicos.csv:datos_hostoricos_meteorologicos:Month
		
		datos_hostoricos_meteorologicos.csv:datos_hostoricos_meteorologicos:Day
			70%

Todos los atributos en los cuales, minimo, el 70% del total de sus valores sean de tipo Entero, su numero minimo de cifras sea igual a 4 y su numero maximo de cifras igual a 4 (YEAR)	SELECT ?nmin ?nmax (?x AS ?Format) (?c AS ?Porcentaje) (?z AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset)
	WHERE {
	?x rdf:type sf:Integer .
	?x sf:percentage ?c.
	?x sf:numCifrasMin ?nmin .
	?x sf:numCifrasMax ?nmax .
	?y sf:haveStorageFormat ?x .
	?z phdd:isDescribedBy ?y .
	?w phdd:column ?z .
	?b phdd:isStructuredBy ?w .
	?b dcterms:title ?f .
	?m dcat:distribution ?b .
	?m dcterms:title ?k .
	FILTER(?c >= 70) .
	FILTER(?nmin = 4 && ?nmax = 4) .
	}
		datos_hostoricos_meteorologicos.csv:datos_hostoricos_meteorologicos:Year	**70%**

Figura 8.2: Consulta sobre el modelo ontológico común: atributo Fecha

Atributo Esquema Mediador:	Hora		
CONSULTAS DE EXTRACCIÓN DE DATOS	**CONSULTA SPARQL**	**ATRIBUTOS RESULTANTES**	**% DATOS ANALIZADOS**
Todos los atributos en los cuales, minimo, el 70% del total de sus valores sean de tipo sf:Date con un formato 'HH:mm'	SELECT (?x AS ?Format) (?c AS ?Porcentaje) (?z AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset) (?a AS ?FormatDate) WHERE { ?x rdf:type sf:Date . ?x sf:percentage ?c. ?x sf:formatDate ?a . ?y sf:haveStorageFormat ?x . ?z phdd:isDescribedBy ?y . ?w phdd:column ?z . ?b phdd:isStructuredBy ?w . ?b dcterms:title ?f . ?m dcat:distribution ?b . ?m dcterms:title ?k . FILTER(?c >= 70) . FILTER(?a = "yyyy-mm-dd") .}	Climatica_Estmarianza.xls:Climatica_Estmarianza05082015:Time datos_climaticos_por_dia.xml/mes:hora:Hora	**70%**
Todos los atributos en los cuales, minimo, el 70% del total de sus valores sean de tipo sf:Date con cualquier tipo de formato	SELECT (?x AS ?Format) (?c AS ?Porcentaje) (?z AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset) (?a AS ?FormatDate) WHERE { ?x rdf:type sf:Date . ?x sf:percentage ?c. ?x sf:formatDate ?a . ?y sf:haveStorageFormat ?x . ?z phdd:isDescribedBy ?y . ?w phdd:column ?z . ?b phdd:isStructuredBy ?w . ?b dcterms:title ?f . ?m dcat:distribution ?b . ?m dcterms:title ?k . FILTER(?c >= 70) .}	• Climatica_Estmarianza.xls:Climatica_Estmarianza05082015:Date • • datos_climaticos_por_dia.xml/mes:dia:Dia • • Climatica_Estmarianza.xls:Climatica_Estmarianza05082015:Time datos_climaticos_por_dia.xml/mes:dia:hora	**70%**
Todos los atributos en los cuales, minimo, el 70% del total de sus valores sean de tipo Entero, su numero minimo sea mayor igual a cero y su numero maximo menor igual a 23 (HORA)	SELECT ?nmin ?nmax (?x AS ?Format) (?c AS ?Porcentaje) (?z AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset) WHERE { ?x rdf:type sf:Integer . ?x sf:percentage ?c. ?x sf:numeroMinimoInteger ?nmin ?x sf:numeroMaximoInteger ?nmax ?y sf:haveStorageFormat ?x . ?z phdd:isDescribedBy ?y . ?w phdd:column ?z . ?b phdd:isStructuredBy ?w . ?b dcterms:title ?f . ?m dcat:distribution ?b . ?m dcterms:title ?k . FILTER(?c >= 70) . FILTER(?nmin >= 0 && ?nmax <=23) }	• datos_historicos_meteorologicos.csv:datos_hosticos_meteorologicos:Month • datos_historicos_meteorologicos.csv:datos_hosticos_meteorologicos:Minute • datos_historicos_meteorologicos.csv:datos_hosticos_meteorologicos:Cloudy • datos_historicos_meteorologicos.csv:datos_hosticos_meteorologicos:Hour datos_historicos_meteorologicos.csv:datos_hosticos_meteorologicos:Historical Record	**70%**
Todos los atributos en los cuales, mínimo, el 70% del total de sus valores sean de tipo Entero, su número mínimo sea mayor igual a cero y su número máximo menor igual a 59 (MINUTO)	SELECT ?nmin ?nmax (?x AS ?Format) (?c AS ?Porcentaje) (?z AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset) WHERE { ?x rdf:type sf:Integer . ?x sf:percentage ?c. ?x sf:numeroMinimoInteger ?nmin ?x sf:numeroMaximoInteger ?nmax ?y sf:haveStorageFormat ?x . ?z phdd:isDescribedBy ?y . ?w phdd:column ?z . ?b phdd:isStructuredBy ?w . ?b dcterms:title ?f . ?m dcat:distribution ?b . ?m dcterms:title ?k . FILTER(?c >= 70) . FILTER(?nmin >= 0 && ?nmax <=59) .}	• datos_historicos_meteorologicos.csv:datos_hosticos_meteorologicos:Month • datos_historicos_meteorologicos.csv:datos_hosticos_meteorologicos:Minute • datos_historicos_meteorologicos.csv:datos_hosticos_meteorologicos:Cloudy • datos_historicos_meteorologicos.csv:datos_hosticos_meteorologicos:Hour • datos_historicos_meteorologicos.csv:datos_hosticos_meteorologicos:Day	

Figura 8.3: Consulta sobre el modelo ontológico común: atributo Hora

Atributo Esquema Mediador:	HUMEDAD		
CONSULTAS DE EXTRACCIÓN DE DATOS	**CONSULTA SPARQL**	**ATRIBUTOS RESULTANTES**	**% DATOS ANALIZADOS**

Todos los atributos en los cuales, mínimo, el 70% del total de sus valores sean de tipo Decimal, su número mínimo de decimales sea mayor o igual a cero y su número máximo de decimales sea menor igual a 2. Su número mínimo sea mayor igual a cero y su número máximo menor igual a 100(**PORCENTAJE DE HUMEDAD)**	SELECT ?nmin ?nmax (?x AS ?Format) (?c AS ?Porcentaje) (?g AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset)
	WHERE {
	?x rdf:type sf:Decimal .
	?x sf:percentage ?c.
	?x sf:numeroMinimo ?nmin .
	?x sf:numeroMaximo ?nmax .
	?x sf:numMaximoDecimales ?nmaxDec .
	?x sf:numMinimoDecimales ?nminDec .
	?y sf:haveStorageFormat ?x .
	?z phdd:isDescribedBy ?y .
	?z skos:prefLabel ?g .
	?w phdd:column ?z .
	?b phdd:isStructuredBy ?w .
	?b dcterms:title ?f .
	?m dcat:distribution ?b .
	?m dcterms:title ?k .
	FILTER(?c >= 70) .
	FILTER(?nminDec >= 0 && ?nmaxDec <= 2) .
	FILTER(?nmin >= 0 && ?nmax <= 100) .
	}
		• datos_climaticos_por_dia.xml/mes:Meteoros:Nivel1.._a_0cm
		• datos_climaticos_por_dia.xml/mes:Meteoros:Nivel2.._a_0cm
		• datos_climaticos_por_dia.xml/mes:Meteoros:pH._a_0cm
		• datos_climaticos_por_dia.xml/mes:Meteoros:Turbidez._a_0cm
		• datos_climaticos_por_dia.xml/mes:Meteoros:Oxigeno._a_0cm
		• datos_climaticos_por_dia.xml/mes:Meteoros:Humedad._a_400cm
		• datos_climaticos_por_dia.xml/mes:Meteoros:Temp.Agua._a_0cm
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:In Heat
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Wind Run
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Processed Record]
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Wind Speed
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Temp Out
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Rain Rate
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Out Cloudy
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:In Temp
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Low Temp
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:HI temp [F]
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:In Hum
			70%

Figura 8.4: Consulta sobre el modelo ontológico común: atributo Humedad

Atributo Esquema Mediador:	NUBOSIDAD		
CONSULTAS DE EXTRACCIÓN DE DATOS	**CONSULTA SPARQL**	**ATRIBUTOS RESULTANTES**	**% DATOS ANALIZADOS**

Todos los atributos en los cuales, mínimo, el 70% del total de sus valores sean de tipo Integer, Su número mínimo sea mayor igual a cero y su número máximo menor igual a 8(**Octal)**	SELECT ?nmin ?nmax (?x AS ?Format) (?c AS ?Porcentaje) (?g AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset)
	WHERE {
	?x rdf:type sf:Integer .
	?x sf:percentage ?c.
	?x sf:numeroMinimoInteger ?nmin .
	?x sf:numeroMaximoInteger ?nmax .
	?y sf:haveStorageFormat ?x .
	?z phdd:isDescribedBy ?y .
	?z skos:prefLabel ?g .
	?w phdd:column ?z .
	?b phdd:isStructuredBy ?w .
	?b dcterms:title ?f .
	?m dcat:distribution ?b .
	?m dcterms:title ?k .
	FILTER(?c >= 70) .
	FILTER(?nmin >= 0 && ?nmax <= 8)
	}
		• datos_hostoricos_meteorologicos.csv:datos_hostoricos_meteorologicos:Historical Record
		• datos_hostoricos_meteorologicos.csv:datos_hostoricos_meteorologicos:Minute
		• datos_hostoricos_meteorologicos.csv:datos_hostoricos_meteorologicos:Cloudy
			70%

Todos los atributos en los cuales, mínimo, el 70% del total de sus valores sean de tipo Decimal, su número mínimo de decimales sea mayor o igual a 1 y su número máximo de decimales sea menor igual a 1. Su número mínimo sea mayor igual a cero y su número máximo menor igual a 8(**OCTAL DECIMALES)**	SELECT ?nmin ?nmax (?x AS ?Format) (?c AS ?Porcentaje) (?g AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset)
	WHERE {
	?x rdf:type sf:Decimal .
	?x sf:percentage ?c.
	?x sf:numeroMinimo ?nmin .
	?x sf:numeroMaximo ?nmax .
	?x sf:numMaximoDecimales ?nmaxDec .
	?x sf:numMinimoDecimales ?nminDec .
	?y sf:haveStorageFormat ?x .
	?z phdd:isDescribedBy ?y .
	?z skos:prefLabel ?g .
	?w phdd:column ?z .
	?b phdd:isStructuredBy ?w .
	?b dcterms:title ?f .
	?m dcat:distribution ?b .
	?m dcterms:title ?k .
	FILTER(?c >= 70) .
	FILTER(?nminDec >= 1 && ?nmaxDec <= 1) .
	FILTER(?nmin >= 0 && ?nmax <= 8) .
	}
		• datos_climaticos_por_dia.xml/mes:Meteoros:Precip.._a_50mm
		• datos_climaticos_por_dia.xml/mes:Meteoros cloud1.._a_0cm
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Wind Speed
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Out Cloudy
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Processed Record	

Figura 8.5: Consulta sobre el modelo ontológico común: atributo Nubosidad

Atributo Esquema Mediador:	PRECIPITACION		
CONSULTAS DE EXTRACCIÓN DE DATOS	**CONSULTA SPARQL**	**ATRIBUTOS RESULTANTES**	**% DATOS ANALIZADOS**
Todos los atributos en los cuales, mínimo, el 70% del total de sus valores sean de tipo Decimal, su número mínimo de decimales sea mayor o igual a cero y su número máximo de decimales sea menor igual a 3. Su número mínimo sea mayor igual a cero y su número máximo menor igual a 7(**milímetros)**	SELECT ?nmin ?nmax (?x AS ?Format) (?c AS ?Porcentaje) (?g AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset) WHERE { ?x rdf:type sf:Decimal . ?x sf:percentage ?c. ?x sf:numeroMinimo ?nmin . ?x sf:numeroMaximo ?nmax . ?x sf:numMaximoDecimales ?nmaxDec ?x sf:numMinimoDecimales ?nminDec . ?y sf:haveStorageFormat ?x . ?z phdd:isDescribedBy ?y . ?z skos:prefLabel ?g . ?w phdd:column ?z . ?b phdd:isStructuredBy ?w . ?b dcterms:title ?f . ?m dcat:distribution ?b . ?m dcterms:title ?k . FILTER(?c >= 70) . FILTER(?nminDec >= 0 && ?nmaxDec <= 3) . FILTER(?nmin >= 0 && ?nmax <= 7) .}	• datos_climaticos_por_dia.xml/mes:Meteoros:Precip.._a_50mm • datos_climaticos_por_dia.xml/mes:Meteoros:cloud1.._a_0cm • datos_climaticos_por_dia.xml/mes:Meteoros:Nivel2.._a_0cm • datos_climaticos_por_dia.xml/mes:Meteoros:Nivel1.._a_0cm • Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Processed Record • Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Wind Speed • Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:In Precipitation [mm]	**70%**
Todos los atributos en los cuales, mínimo, el 70% del total de sus valores sean de tipo Decimal, su número mínimo de decimales sea mayor o igual a cero y su número máximo de decimales sea menor igual a 3. Su número mínimo sea mayor igual a cero y su número máximo menor igual a 0.7(**centimetros)**	SELECT ?nmin ?nmax (?x AS ?Format) (?c AS ?Porcentaje) (?g AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset) WHERE { ?x rdf:type sf:Decimal . ?x sf:percentage ?c. ?x sf:numeroMinimo ?nmin . ?x sf:numeroMaximo ?nmax . ?x sf:numMaximoDecimales ?nmaxDec . ?x sf:numMinimoDecimales ?nminDec . ?y sf:haveStorageFormat ?x . ?z phdd:isDescribedBy ?y . ?z skos:prefLabel ?g . ?w phdd:column ?z . ?b phdd:isStructuredBy ?w . ?b dcterms:title ?f . ?m dcat:distribution ?b . ?m dcterms:title ?k . FILTER(?c >= 70) . FILTER(?nminDec >= 0 && ?nmaxDec <= 3) . FILTER(?nmin >= 0 && ?nmax <= 0.7).}	• datos_hostoricos_meteorologicos.csv:datos_hostoricos_meteorologicos:Total Precipitation [cm] • datos_climaticos_por_dia.xml/mes:Meteoros:Nivel2.._a_0cm • datos_climaticos_por_dia.xml/mes:Meteoros:Nivel1.._a_0cm • Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:In Precipitation [cm]	**70%**

Figura 8.6: Consulta sobre el modelo ontológico común: atributo Precipitación

Atributo Esquema Mediador:	PROCESADO		
CONSULTAS DE EXTRACCIÓN DE DATOS	**CONSULTA SPARQL**	**ATRIBUTOS RESULTANTES**	**% DATOS ANALIZADOS**

Todos los atributos en los cuales, mínimo, el 70% del total de sus valores sean de tipo Boolean	SELECT ?nmin ?nmax (?x AS ?Format) (?c AS ?Porcentaje) (?g AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset)
	WHERE {
	?x rdf:type sf:Boolean .
	?x sf:percentage ?c.
	?x sf:numeroMinimo ?nmin .
	?x sf:numeroMaximo ?nmax .
	?x sf:numMaximoDecimales ?nmaxDec .
	?x sf:numMinimoDecimales ?nminDec .
	?y sf:haveStorageFormat ?x .
	?z phdd:isDescribedBy ?y .
	?z skos:prefLabel ?g .
	?w phdd:column ?z .
	?b phdd:isStructuredBy ?w .
	?b dcterms:title ?f .
	?m dcat:distribution ?b .
	?m dcterms:title ?k .
	FILTER(?c >= 70) .
	}
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Processed Record	**70%**

Todos los atributos en los cuales, mínimo, el 70% del total de sus valores sean de tipo Integer, Su número mínimo sea mayor igual a cero y su número máximo menor igual a 1(**Octal**)	SELECT ?nmin ?nmax (?x AS ?Format) (?c AS ?Porcentaje) (?g AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset)
	WHERE {
	?x rdf:type sf:Integer .
	?x sf:percentage ?c.
	?x sf:numeroMinimoInteger ?nmin .
	?x sf:numeroMaximoInteger ?nmax .
	?y sf:haveStorageFormat ?x .
	?z phdd:isDescribedBy ?y .
	?z skos:prefLabel ?g .
	?w phdd:column ?z .
	?b phdd:isStructuredBy ?w .
	?b dcterms:title ?f .
	?m dcat:distribution ?b .
	?m dcterms:title ?k .
	FILTER(?c >= 70) .
	FILTER(?nmin >= 0 && ?nmax <= 1) .
	}
		• datos_hostoricos_meteorologicos.csv:datos_hostoricos_meteorologicos:Historical Record
		• datos_hostoricos_meteorologicos.csv:datos_hostoricos_meteorologicos:Minute
			70%

Figura 8.7: Consulta sobre el modelo ontológico común: atributo Procesado

Atributo Esquema Mediador:	TEMPERATURA		
CONSULTAS DE EXTRACCIÓN DE DATOS	**CONSULTA SPARQL**	**ATRIBUTOS RESULTANTES**	**% DATOS ANALIZADOS**
Todos los atributos en los cuales, mínimo, el 70% del total de sus valores sean de tipo Decimal, su número mínimo de decimales sea mayor o igual a cero y su número máximo de decimales sea menor igual a 2. Su número mínimo sea mayor igual a cero y su número máximo menor igual a 35(**Grados centigrados)**	SELECT ?nmin ?nmax (?x AS ?Format) (?c AS ?Porcentaje) (?g AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset) WHERE { ?x rdf:type sf:Decimal . ?x sf:percentage ?c. ?x sf:numeroMinimo ?nmin . ?x sf:numeroMaximo ?nmax . ?x sf:numMaximoDecimales ?nmaxDec . ?x sf:numMinimoDecimales ?nminDec . ?y sf:haveStorageFormat ?x . ?z phdd:isDescribedBy ?y . ?z skos:prefLabel ?g . ?w phdd:column ?z . ?b phdd:isStructuredBy ?w . ?b dcterms:title ?f . ?m dcat:distribution ?b . ?m dcterms:title ?k . FILTER(?c >= 70) . FILTER(?nminDec >= 0 && ?nmaxDec <= 2) . FILTER(?nmin >= 0 && ?nmax <= 35) . }	• datos_climaticos_por_dia.xml/mes: Meteoros:Nivel1.._a_0cm • datos_climaticos_por_dia.xml/mes: Meteoros:Nivel2.._a_0cm • datos_climaticos_por_dia.xml/mes: Meteoros:pH.._a_0cm • datos_climaticos_por_dia.xml/mes: Meteoros:Turbidez.._a_0cm • Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:In Heat • Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:Wind Run • Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:Processed Record] • Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:Low Temp • Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:Temp Out • Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:Rain	**70%**

Todos los atributos en los cuales, mínimo, el 70% del total de sus valores sean de tipo Decimal, su número mínimo de decimales sea mayor o igual a cero y su número máximo de decimales sea menor igual a 2. Su número mínimo sea mayor igual a cero y su número máximo menor igual a 95(**Grados Farengheit)**	SELECT ?nmin ?nmax (?x AS ?Format) (?c AS ?Porcentaje) (?g AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset)
	WHERE {
	?x rdf:type sf:Decimal .
	?x sf:percentage ?c.
	?x sf:numeroMinimo ?nmin .
	?x sf:numeroMaximo ?nmax .
	?x sf:numMaximoDecimales ?nmaxDec .
	?x sf:numMinimoDecimales ?nminDec .
	?y sf:haveStorageFormat ?x .
	?z phdd:isDescribedBy ?y .
	?z skos:prefLabel ?g .
	?w phdd:column ?z .
	?b phdd:isStructuredBy ?w .
	?b dcterms:title ?f .
	?m dcat:distribution ?b .
	?m dcterms:title ?k .
	FILTER(?c >= 70) .
	FILTER(?nminDec >= 0 && ?nmaxDec <= 2) .
	FILTER(?nmin >= 0 && ?nmax <= 95) .
	}
		• datos_climaticos_por_dia.xml/mes: Meteoros:Nivel1.._a_0cm
		• datos_climaticos_por_dia.xml/mes: Meteoros:Nivel2.._a_0cm
		• datos_climaticos_por_dia.xml/mes: Meteoros:pH.._a_0cm
		• datos_climaticos_por_dia.xml/mes: Meteoros:Turbidez.._a_0cm
		• Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:In Heat
		• Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:Wind Run
		• Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:Processed Record]
		• Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:HI Temp[F]
		• Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:Temp Out
		• Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:Rain
		• Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:Out Cloudy
		• Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:In Temp
		• Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:Low Temp
		• Climatica_Estmarianza.xls:Climatica _Estamarianza05082015:HI temp [C]
			70%

Figura 8.8: Consulta sobre el modelo ontológico común: atributo Temperatura

Atributo Esquema Mediador:	VIENTO		
CONSULTAS DE EXTRACCIÓN DE DATOS	**CONSULTA SPARQL**	**ATRIBUTOS RESULTANTES**	**% DATOS ANALIZADOS**

Todos los atributos en los cuales, mínimo, el 70% del total de sus valores sean de tipo Decimal, su número mínimo de decimales sea mayor o igual a cero y su número máximo de decimales sea menor igual a 3. Su número mínimo sea mayor igual a cero y su número máximo menor igual a 20(**km/h**)	SELECT ?nmin ?nmax (?x AS ?Format) (?c AS ?Porcentaje) (?g AS ?Columna) (?f AS ?Tabla) (?k AS ?Dataset)
	WHERE {
	?x rdf:type sf:Decimal .
	?x sf:percentage ?c.
	?x sf:numeroMinimo ?nmin .
	?x sf:numeroMaximo ?nmax .
	?x sf:numMaximoDecimales ?nmaxDec .
	?x sf:numMinimoDecimales ?nminDec .
	?y sf:haveStorageFormat ?x .
	?z phdd:isDescribedBy ?y .
	?z skos:prefLabel ?g .
	?w phdd:column ?z .
	?b phdd:isStructuredBy ?w .
	?b dcterms:title ?f .
	?m dcat:distribution ?b .
	?m dcterms:title ?k .
	FILTER(?c >= 70) .
	FILTER(?nminDec >= 0 && ?nmaxDec <= 3) .
	FILTER(?nmin >= 0 && ?nmax <= 20) .
	}
		• datos_climaticos_por_dia.xml/mes:Meteoros:Nivel2.._a_0cm
		• datos_climaticos_por_dia.xml/mes:Meteoros:Nivel1.._a_0cm
		• datos_climaticos_por_dia.xml/mes:Meteoros:pH.._a_0cm
		• datos_climaticos_por_dia.xml/mes:Meteoros:Turbidez.._a_0cm
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Wind Speed
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Wind Chill
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Processed Record]
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Low Temp
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Temp Out
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:Out Cloudy
		• Climatica_Estmarianza.xls:Climatica_Estamarianza05082015:HI Temp[C]
		• datos_hostoricos_meteorologicos.csv:datos_hostoricos_meteorologicos:Wind speed [10 m above gnd]
		• datos_hostoricos_meteorologicos.csv:datos_hostoricos_meteorologicos:Wind speed [80 m above gnd]
		• datos_hostoricos_meteorologicos.csv:datos_hostoricos_meteorologicos:Wind speed [900 mb]
		• datos_hostoricos_meteorologicos.csv:datos_hostoricos_meteorologicos:Total Precipitation [m]
			70%

Figura 8.9: Consulta sobre el modelo ontológico común: atributo Viento

ANEXO H. Asignaciones semánticas obtenidas de manera manual y mediante el modelo ontológico común

EL ANEXO H, se adjunta de manera digital bajo el nombre ANEXOH.pdf, para una mejor visualización del archivo, debido a que se trata de una archivo bastante grande.

ANEXO I. Descripción de fuentes de datos de ejemplo (Descubrimiento de asignaciones semánticas)

Campo	Tipo de Dato	Descripción	Unidades de Medida
In Dew	Decimal	Describe el punto de rocío interior	F
In Heat	Decimal	Índice de calor interno; incorpora humedad en la temperatura	F
Wind Samp	Decimal	número de muestras durante el intervalo de archivo	F
Wind Tx	Entero	conexión; Trabaja = 1, no funciona = 0	1 o 0
In Precipitation [mm]	Decimal	El valor de precipitación en mm se refiere a la cantidad de lluvia por metro cuadrado en una hora	mm
In Precipitation [cm]	Decimal	El valor de precipitación en cm se refiere a la cantidad de lluvia por metro cuadrado en una hora	cm
Processed Record	Booleano	Describe el porcentaje de datos procesados	%

Cuadro 8.25: Descripción de la fuente S4, tipo XLS(Climatica_Estmarianza.xls)

Campo	Tipo de Dato	Descripción	Unidades de Medida
precipitacion	decimal	Describe el depósito de agua en superficie de la Tierra, en forma de lluvia, nieve, hielo o granizo	Milímetros(mm)
nubosidad	decimal	Describe la fracción de cielo cubierto con nubes, en un lugar en particular	OCTA
evaporación	decimal	experimenta una sustancia a partir de un estado líquido a un estado de vapor o gas	milímetro (mm)
temperatura	decimal	magnitud que mide el nivel térmico o el calor que un cuerpo posee	Kelvin (K)
viento	decimal	En las mediciones del viento se especifica su intensidad o fuerza (unidad = m/s) y su dirección	metros por segundo (m/s)
fecha	fecha	Describe el momento en la cual se registro la información	YYYY-MM-DD HH:mm:ss
heliofania	decimal	Tiempo de duración del brillo solar. Se mide en horas y minutos de brillo solar	

Cuadro 8.26: Descripción de la fuente S1, tipo Base de Datos(clima.sql)

Campo	Tipo de Dato	Descripción	Unidades de Medida
Fecha	Fecha	Describe el momento en la cual se registró la información	YYYY-MM-DD
Hora	Fecha	Describe la hora en la cual se registró la información	HH:mm
Caudal1	Decimal	Cantidad de agua que lleva una corriente o que fluye de un manantial o fuente	metros cúbicos por segundo
Conduct	Decimal	Describe el valor inverso de la resistencia y se mide como la cantidad de conductancia en una distancia determinada	mhos/cm
Humedad	Decimal	Es la cantidad de vapor de agua que se encuentra por unidad de volumen de aire de un ambiente	g/m3 = gramos de agua por cada metro cúbico de aire
Oxigeno	Decimal	cantidad de oxigeno gaseoso que está disuelto en el agua	mg/L
Precip	decimal	Describe el depósito de agua de la superficie de la Tierra, en forma de lluvia, nieve, hielo o granizo	Milímetros(mm)
Turbidez	Decimal	Representa el grado en el cual el agua pierde su transparencia debido a la presencia de partículas en suspensió	UNT(Nephelometric Turbidity Unit)

Cuadro 8.27: Descripción de la fuente S3, tipo de XML(datosClimaticosPorDia.xml)

Campo	Tipo de Dato	Descripción	Unidades de Medida
Temperature	Decimal	magnitud que mide el nivel térmico o el calor que un cuerpo posee	Kelvin (K)
Relative humidity	Decimal	cantidad de humedad en el aire en comparación con lo que el aire puede retener.[a] esa temperatura	
Mean Sea Level Pressure	Decimal	Define la presión media del nivel del mar	
Total Precipitation	Decimal	El valor de precipitación en mm se refiere a la cantidad de lluvia por metro cuadrado en una hora	mm
Total cloud cover	Decimal	refiere a la fracción del cielo cubierto por nubes de un tipo o combinación en particular	
Wind speed	Decimal	Describe la velocidad del movimiento del aire en un entorno exterior	m/s
Year	Entero	Describe el año en la cual se registró la información	
Month	Entero	Describe el mes en la cual se registró la información	
Day	Entero	Describe el día en la cual se registró la información	
Minute	Entero	Describe el minuto en la cual se registró la información	

Cuadro 8.28: Descripción de la fuente S2, tipo CSV(descripcionDeEsquemaMediador.csv)

ANEXO J. Publicación del Archivo RDF

En este apartado se procede a publicar los datos RDF con el objetivo de que usuarios puedan acceder a esta información mediante consultas SPARQL. Para ello existe servidores de datos RDF o llamados también servidor de tripletas, que la principal función es gestionar la información de los archivos RDF.

Proceso de publicación

Figura A: Proceso de Publicación de un RDF

En esta sesión procederemos a instalar y configurar Marmotta, se seleccionó la versión 3.3.0 que es la última versión disponible. Soporta estándares de GEOSPARQL, incluye SPARQL server además tiene conjunto de módulos y bibliotecas para la creación de aplicaciones de datos personalizadas vinculadas.

Esta herramienta proporciona librerías que se pueden ser usados fuera de la plataforma Marmotta, por ejemplo existe librerías que permite el acceso a los recursos.

En la figura A se observa el proceso aplicado para la publicación de un archivo RDF.

Paso para el proceso de publicación

1. **Instalación y Configuración del Servidor de Tripletas:** Marmota nos proporciona diversas formas de instalar. Para este caso se seleccionó la instalación binaria que básicamente es descargar un archivo con extensión .war y desplegar en cualquier servidor de aplicaciones.

2. **Carga de los archivos RDF al servidor de Tripletas** Con el objetivo de no tener inconvenientes al momento de publicar o al hacer consultas, se procedió a validar el archivo RDF, permitiendo así tener un archivo sin errores de sintaxis.
   Existen dos formar de cargar archivos al servidor, la primera mediante la plataforma web,donde se selecciona el archivo RDF y presionamos en subir(Figura B).

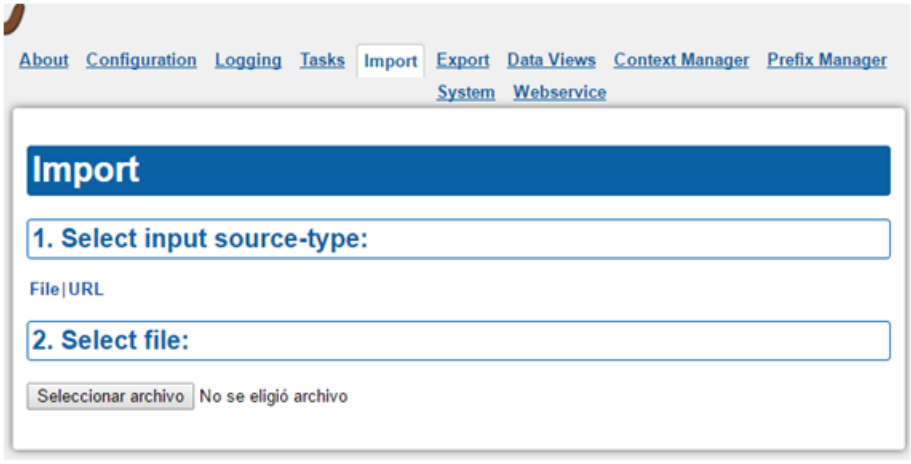


Figura B: Subir RDF manualmente

La segunda forma es usando uno de los componentes que nos proporciona Marmotta, se tiene que especificar la ruta física del archivo RDF y los parámetros de conexión al servidor(Figura C).

```
String path = "C:\\Users\\Pablo\\ontologias\\result.rdf";
String context = "http://example.org/context";
ClientConfiguration configuration = new ClientConfiguration("http://localhost/marmotta/","admin","1234");
ImportClient importClient = new ImportClient(configuration);
InputStream is = new FileInputStream(new File(path));
RDFFormat format = Rio.getParserFormatForFileName(path);
importClient.uploadDataset("none", format.getDefaultMIMEType(), context);
```

Figura C: Método que Permite subir un RDF

Figura D: RDF disponible en la plataforma

En la Figura D se observa el archivo RDF publicado y listo para ser explotado mediante consultas SPARQL.

Bibliografía

[1] Bizer, "Server–publishing relational databases on the semantic web", Poster at the Web Conference,2006.

[2] AnHai Doan,Alon Halevy, Zachary Ives, "Principles of Data Integration. 225 Wyman Street, Waltham", Poster at the Web Conference,MA 02451, USA. 6-10.

[3] AnHai Doan,Alon Halevy, Zachary Ives, "Principles of Data Integration. 225 Wyman Street, Waltham", Poster at the Web Conference,MA 02451, USA. 1-20.

[4] AnHai Doan,Alon Halevy, Zachary Ives, "Principles of Data Integration. 225 Wyman Street, Waltham", Poster at the Web Conference,MA 02451, USA. 10-17.

[5] World Wide Web Consortium (W3C), " WEB SEMANTICA", World Wide Web Consortium (W3C),http://www.w3c.es/Divulgacion/GuiasBreves/WebSemantica.

[6] The Open Source Geospatial Foundation, " GeoTools User Guide. The Open", Source Geospatial Foundation,http://docs.geotools.org/latest/userguide/.

[7] World Wide Web Consortium (W3C), " Resource Description Framework (RDF)", World Wide Web Consortium (W3C), https://www.w3.org/RDF/

[8] J. Wang, Z. Miao, " Semantic integration of relational data using SPARQL", International Symposium on Intelligent Information Technology Application,2008.

[9] J. Wang, Z. Miao, " Semantic integration of relational data using SPARQL", International Symposium on Intelligent Information Technology Application,2008.

[10] World Wide Web Consortium (W3C), "Physical Data Description (PHDD)",http://rdf-vocabulary.ddialliance.org/phdd.html

[11] Apache Jena, “Apache Jena”,https://jena.apache.org/

[12] World Wide Web Consortium (W3C), ”Lenguaje de Ontologías Web (OWL)”,https://www.w3.org/TR/rdf-sparql-query/

[13] World Wide Web Consortium (W3C), ”SPARQL Query Language for RDF”,https://www.w3.org/2007/09/OWL-Overview-es.html

[14] CSVReader, “ Reader for CSV”,https://www.csvreader.com/

[15] The Apache Software Foundation,”Maven”http://maven.apache.org/what-is-maven.html

[16] Apache Software Foundation, .ªpache POI - the Java API for Microsoft Documents”,https://poi.apache.org/

[17] Apache Marmotta, “Apache Marmotta”,http://marmotta.apache.org/index.html

[18] Apache Tika, “Apache ”, https://tika.apache.org/

[19] Java NIO, “Java”, http://tutorials.jenkov.com/java-nio/index.html

[20] Java IO, “Java”, https://docs.oracle.com/javase/tutorial/essential/io/index.html

[21] ESRI, “Shapefile Technical Description”, ESRI White Paper,July 1998

[22] M.I.D. Norma Laura Salazar Viveros, “Administración de base de Datos”,2014-05-22

[23] Creativyst Software,Ünderstanding CSV File Formats”, http://www.creativyst.com/Doc/Articles/CSV/CSV01.htm

[24] Pendiente proceso de integracion

[25] Silberschatz, Abraham. McGRAW-HILL, ed. Fundamentos de bases de datos.

[26] James Gosling, Bill Joy, Guy Steele, y Gilad Bracha, The Java language specification, tercera edición. Addison-Wesley, 2005.

[27] Universidad de Valladolid, Ïntrodución al SQL”, https://www.infor.uva.es/ jvegas/cursos/bd/sqlplus/sqlplus.html, Abril de 1998

[28] Cristina del Pino, Elsa Aguado (2012). Internet, Televisión y Convergencia: nuevas pantallas y plataformas de contenido audiovisual en la era digital , 19 de enero de 2016.

[29] Ayers, Danny; Völkel, 3 Diciembre 2008. Çool URIs for the Semantic Web”. World Wide Web Consortium.

[30] World Wide Web Consortium, "W3C", https://www.w3c.com/

[31] World Wide Web Consortium (W3C), " DDI-RDF Discovery Vocabulary (Disco)",http://www.ddialliance.org/Specification/RDF/Discovery

[32] World Wide Web Consortium (W3C), "XML Query Language",https://www.w3.org/TR/xquery/

[33] World Wide Web Consortium (W3C), " DDI-RDF Discovery Vocabulary (Disco)",http://www.ddialliance.org/Specification/RDF/Discovery

[34] World Wide Web Consortium (W3C), " Data Catalog Vocabulary (DCAT)",https://www.w3.org/TR/vocab-dcat/

[35] DDI RDF Vocabularies, https://www.ddialliance.org/Specification/RDF

[36] Caridad Anías Calderón, Julio Cesar Jerez Camps, Yinett Mazón Fernández, Joan Manuel Granadillo,APROXIMACIONES A LA INTEGRACIÓN DE INFORMACIÓN Y APLICACIONES, Vol. 11. No. 3,diciembre, 2012.

[37] UNIVERSIDAD DE CUENCA,"PROMAS", https://www.ucuenca.edu.ec/la-investigacion/unidades-de-investigacion/promas

[38] JDOM Project, "JDOM", http://jdom.org

[39] Dbpedia, "Sgvizler", http://wiki.dbpedia.org/projects/sgvizler

[40] Oracle, https://docs.oracle.com/en/database/oracle/oracle-database/12.2/index.html

[41] Mysql, https://dev.mysql.com/doc/

[42] Postgres, https://www.postgresql.org/docs/

[43] Biblioteca Meteorológica Nacional, "The Beaufort Scale". Met Office, 02-10-2012, https://es.scribd.com/document/370317842/Beaufort-Scale