

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADUADO EN INGENIERÍA INFORMÁTICA

Análisis de la Actividad de los Alumnos en el Campus Virtual

Analysis of Student Activity in the Virtual Campus

Realizado por
Manuel Paneque Romero

Tutorizado por
María del Mar Roldán García

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, SEPTIEMBRE de 2019

Fecha defensa: Septiembre de 2019

Fdo. El/la Secretario/a del Tribunal

Agradecimientos

El presente trabajo fin de grado fue realizado bajo la supervisión de María del Mar Roldán García, a la cual quiero agradecer la confianza depositada en mí al asignarme este trabajo y su implicación en su desarrollo. Gracias por guiarme durante todo este proceso y por compartir tus conocimientos sobre la web semántica.

Quisiera agradecer a José Manuel García Nieto y Javier del Ser Lorente sus aportaciones en conocimientos de machine learning.

Agradecer también a los miembros del grupo de investigación Khaos y en especial a José Francisco Aldana Montes, Antonio Benítez Hidalgo y Sandro Hurtado Requena su apoyo moral y técnico en el día a día.

Gracias al personal docente e Investigador de la Escuela Técnica Superior de Ingeniería Informática, que a lo largo de estos años me han formado y sin los que tampoco hubiera sido posible llegar hasta aquí. También quiero hacer mención al personal de administración y servicios por hacerme sentir como en casa.

Por último, quiero agradecer a mi padre, a mi madre y demás miembros de la familia, su apoyo emocional y económico durante estos años de grado, facilitándome la vida para centrarme en estudiar.

Para finalizar, quisiera dedicar este trabajo a mi abuela Pepita, que desde allá donde esté, sigue iluminando mi vida.

Resumen

En el conjunto de las plataformas de gestión online de la docencia o LMS (Learning Management System), Moodle es posiblemente la más extendida a nivel mundial por las facilidades que ofrece en sus dimensiones organizativa, pedagógica y tecnológica. La Universidad de Málaga no es una excepción, ya que viene utilizando Moodle desde hace años para la integración de su Campus Virtual (CV). Esta plataforma, en su funcionamiento cotidiano, está recolectando grandes cantidades de datos (sobre 20.000 registros diarios) con información sobre las interacciones de los estudiantes, como: entradas, intervenciones en foros, entrega de tareas, consultas, selección de cursos, etc. Moodle ofrece módulos con capacidades de filtrado y estadísticas básicas, aunque la información generada se dispone en informes y ficheros con tablas de datos en crudo, sin posibilidad de obtener conocimiento explícito y significativo sobre la evolución de los alumnos y los cursos. La extracción de dicha información y su transformación en conocimiento útil para la mejora de los procesos educativos es una tarea compleja, ya que requiere de técnicas avanzadas de análisis de datos que den soporte a los educadores. En este proyecto, basándose en técnicas y herramientas de minería de datos, se pretenden generar una serie de funcionalidades para el análisis avanzado de los datos obtenidos a partir de la plataforma Moodle del Campus Virtual. El objetivo es alinear hábitos de uso de CV con desarrollo de los alumnos (rendimiento en las asignaturas) para detectar patrones y tratar de establecer mecanismos de detección, atención y acción temprana.

Palabras clave:

E-learning, Moodle, OWL, RDF, SPARQ, Machine Learning

Abstract

In the group of online Learning Management System (LMS), Moodle is possibly the most widespread worldwide because of the facilities offered in its organizational, pedagogical and technological dimensions. The University of Malaga is no exception, since it has been using Moodle for years to integrate its "Campus Virtual" (CV). This platform, in its daily operation, is collecting large amounts of data (over 20,000 daily records) with information on student interactions, such as: entries, forum interventions, homework assignments, quizzes, course election, etc. Moodle offers modules with filtering capabilities and basic statistics, although the information generated is available in reports and files with raw data tables, without the possibility of obtaining explicit and meaningful knowledge about the evolution of students and courses. The extraction of such information and its transformation into useful knowledge for the improvement of educational processes is a complex task, since it requires advanced data analysis techniques that support educators. In this project, based on data mining techniques and specific tools for this propose, it is expecting to generate a series of functionalities for the advanced analysis of the data obtained from the Moodle platform of the "Campus Virtual". The objective is to align CV use habits with student progress (subject performance) to detect patterns and try to establish mechanisms for detection, attention and early action.

Keywords:

E-learning, Moodle, OWL, RDF, SPARQ, Machine Learning

Índice general

Resumen	I
Abstract.....	II
Introducción	1
1.1. Motivación.....	1
1.2. Objetivos	2
1.3. Propuesta tecnológica	2
1.4. Estructura de la memoria	4
Conocimientos relacionados	5
2.1. E-learning.....	5
2.2. Web Semántica	6
2.3. Ontologías	6
2.3.1 OWL.....	6
2.4. RDF	8
2.5. SPARQL.....	9
2.6. Stardog	10
2.7. Moodle.....	10
Repositorio RDF	15
3.1. Diseño de la ontología	15
3.2. Extracción de los datos del SGBD relacional	20
3.3. Alineamiento de los resultados de las consultas SQL a RDF	21
3.4. Diseño de las consultas en SPARQL necesarias	25
Análisis de datos.....	29
4.1. Análisis temporal.....	29
4.2. Análisis de grupos de usuarios	31
4.3. API.....	35
Conclusiones.....	37
Bibliografía	39
Anexo	41

1

Introducción

La hipótesis principal de este trabajo es, estudiar la integración de todos los datos que se obtienen del campus virtual, así como de otras fuentes de datos externas. Esta integración permite recopilar y fusionar dichos datos de una forma coherente, para poder facilitar su análisis en un futuro. Para realizar dicha integración se utilizarán tecnologías de la web semántica (Semantic Web, 2015), y concretamente, diseñaremos una ontología (OWL 2 Web Ontology Language, 2012). Estos datos integrados servirán de entrada a los algoritmos de análisis. Este TFG se enmarca en el Proyecto de Innovación Educativa de la UMA PIE17-166.

1.1. Motivación

Actualmente las plataformas de gestión online de la docencia se encuentran ampliamente desplegadas en la mayoría de los niveles educativos. Esto se debe fundamentalmente al avance en el acceso a las nuevas tecnologías por parte de la mayoría de la sociedad. Estas plataformas almacenan gran cantidad de información de los usuarios, lo cual nos permite realizar análisis y extraer conclusiones, por ejemplo: cómo el uso de la plataforma se relaciona con unos niveles de rendimiento, estableciendo un perfil de comportamiento de los usuarios. De esta forma, se ofrece al docente herramientas con acceso a nuevos conocimientos que, junto con su experiencia, puedan ayudar a atender de forma temprana y con mayor exactitud a sus alumnos.

En el caso de la Universidad de Málaga se usa el Campus Virtual que implementa Moodle como plataforma de gestión para la docencia. Este sistema genera unos informes con estadísticas básicas que son mostradas en tablas con datos crudos, sin posibilidad de obtener conocimiento explícito y significativo de la evolución de los alumnos.

La extracción de conocimiento partiendo de los datos que contiene la base de datos de Moodle, y su transformación en información útil para mejorar los procesos educativos es una tarea compleja. La complicación reside en que se requieren técnicas avanzadas de análisis de datos que den soporte a los educadores.

De lo descrito anteriormente nace la necesidad de definir una infraestructura analítica específica para medir el rendimiento de los alumnos aprovechando el gran volumen de datos que se está generando la plataforma Campus Virtual, sacándole el

máximo provecho posible, alineando hábitos de uso del Campus virtual con el rendimiento de los alumnos en las asignaturas. Este conocimiento obtenido de la base de datos del Campus Virtual se puede enriquecer aún más integrando diferentes fuentes de datos relacionadas, con diversidad de formatos, diferentes lenguajes de consulta y distintas formas lógicas de representar las relaciones.

Este sistema analítico permite al docente detectar un rendimiento oscilante o variable de los alumnos, pudiendo establecer mecanismo de detección, atención y acción temprana.

1.2. Objetivos

La hipótesis principal de este trabajo es construir una infraestructura de análisis con los datos que recopila el Campus Virtual, enriqueciéndolos a través de diferentes fuentes de datos, con diversidad de formatos, diferentes lenguajes de consulta y distintas formas lógicas de representar las relaciones.

Para ello se pretende diseñar una ontología que servirá como esquema semántico común y guiará el proceso de integración. Como resultado, se generan datos enriquecidos y anotados semánticamente que serán usados posteriormente para visualizar estadísticas básicas, como, por ejemplo, series temporales de las visitas de un alumno o también la serie temporal de las entregas que realiza.

Con estos datos se entrenará un modelo predictor de forma que nos servirá para identificar el perfil al que pertenece un nuevo estudiante. Para establecer el perfil asociado a un alumno, se usará un algoritmo de agrupación (aprendizaje no supervisado), el cual sirve para categorizar datos no etiquetados.

1.3. Propuesta tecnológica

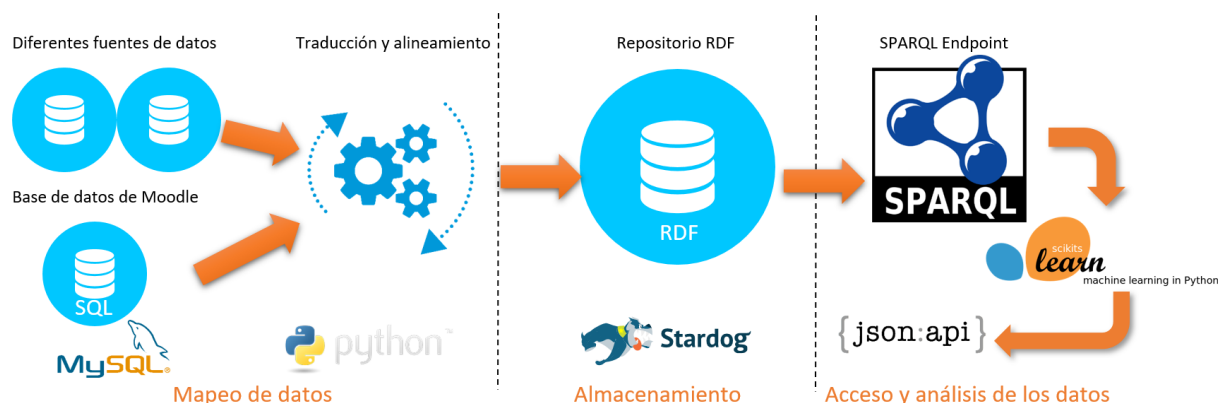


Figura 1: Diagrama de la propuesta tecnológica.

Las principales tecnologías utilizadas para el diseño de la infraestructura de analítica avanzada de la actividad de los usuarios en la plataforma Campus Virtual se expondrán a continuación, realizando una breve introducción a cada una de ellas:

- MySQL (MySQL 8.0 Reference Manual, 2019), se trata de un sistema gestor de base de datos relacional ampliamente utilizado, actualmente propiedad de Oracle. Se va a usar ya que el volcado de los datos originales se encuentran en una base de datos relacional.
- Protégé, (Protégé wiki, 2019) es un programa comúnmente utilizado para diseñar ontologías de código abierto desarrollado por la Universidad de Stanford.
- Stardog (Stardog, 2019), se trata de un sistema gestor de base de datos semántico que soporta los estándares de la Web Semántica: RDF (Hayes & Patel-Schneider, 2014) y OWL (OWL 2 Web Ontology Language, 2012). Stardog soporta consultas en el lenguaje SPARQL (SPARQL 1.1 Overview, 2013). En el capítulo 2 se describirán estas tecnologías con más detalle. Los principales beneficios de Stardog son que permite consultas y transacciones extremadamente rápidas.
- Python, (The Python Language Reference, 2019) es un lenguaje de programación, que tiene una sintaxis de fácil interpretación, con una curva de aprendizaje rápida. Cuenta con una gran cantidad de librerías entre las cuales se han usado en este proyecto: Scikit-learn (Documentation of scikit-learn, 2019) para la parte de aprendizaje máquina y matplotlib (matplotlib user's guide, 2019) para la representación gráfica, así como diversos conectores con las bases de datos usadas en este proyecto, MySQL y Stardog; también se ha usado pandas ya que permite un manejo fácil de los datos y una visualización intuitiva.

La Figura 1 muestra el esquema general de nuestra propuesta. Los datos del Moodle están almacenados en una base de datos MySQL proporcionada por el servicio de Enseñanza Virtual de la UMA, que contiene los datos anonimizados de las interacciones que realizan los usuarios del Grado de informática. Con Python se han implementado las funciones que traducen estos datos a RDF usando la ontología diseñada y que almacenan dichos datos RDF en Stardog. Se han implementado las consultas SPARQL correspondientes para extraer la información necesaria para los algoritmos de machine learning. Finalmente, los resultados de estos análisis se pueden extraer mediante una API. Esta API devuelve los datos en formato JSON, facilitando su integración con otras aplicaciones.

Al tener definida una ontología y los datos integrados en RDF, se pueden añadir datos de otras fuentes de datos RDF directamente, o de fuentes de datos en otros formatos, mediante funciones de traducción a RDF.

1.4. Estructura de la memoria

La estructura de la memoria se ha elaborado en orden cronológico de las fases del diseño:

- Capítulo 2, Conocimientos relacionados, donde se adquieren las bases necesarias para el entendimiento de las diferentes tecnologías usadas.
- Capítulo 3, Repositorio RDF, se describe la infraestructura y mapeos necesarios para crear el repositorio RDF. Este repositorio almacenará los datos necesarios para realizar el análisis.
- Capítulo 4, Análisis de datos, en esta sección se detalla el análisis realizado a los datos, así como las gráficas generadas. Además, se aborda la implementación de una API para devolver la información más relevante de los análisis.
- Capítulo 5, Conclusiones, se tratarán los aspectos más relevantes del trabajo. Asimismo, argumentaremos un apartado de conclusiones personales, donde se reflejan las competencias adquiridas durante este proceso.
- Capítulo 6, Bibliografía de las fuentes consultadas.
- Capítulo 7, Anexo con información detallada de la base de datos relacional de Moodle.

2

Conocimientos relacionados

En este capítulo vamos a describir los distintos conceptos que son necesarios para la correcta comprensión del trabajo realizado, como son el concepto de e-learning, las tecnologías de la Web Semántica OWL, RDF y SPARQL y la estructura de la base de datos relacional de Moodle.

2.1. E-learning

Las técnicas de e-learning (e-Learning, 2007) están en auge ya que con los cambios introducidos por las titulaciones de grado se hace necesario un seguimiento de la evaluación continua del alumnado que en la mayoría de los casos consiste en tareas que se realizan fuera del horario de asistencia presencial, que requieren ser entregadas con posterioridad y evaluadas por el docente. Este es el contexto perfecto para la aplicación del e-learning.

Se entiende por e-learning todos aquellos procesos de aprendizaje que se lleven a cabo a través de Internet, caracterizados por seguir una estructura similar al sistema tradicional conectando a profesores y estudiantes de manera asíncrona para llevar a cabo un proceso de aprendizaje didáctico (e-Learning. Definición y Características, 2007).



Figura 2:Proceso e-learning. Figura basada en (García, 2015)

La Figura 2 muestra las características del proceso de e-learning. El tutor guía el aprendizaje de forma remota, facilita la independencia espacio temporal al poder acceder desde cualquier lugar y a cualquier hora, los contenidos que forman parte del proceso formativo presentan una gran diversidad de formatos electrónicos, permite diversas metodologías de trabajo ya sea individual o en grupos, el alumno puede controlar su proceso de aprendizaje dedicando más horas al contenido que crea necesario, para finalizar permite una comunicación bidireccional entre los estudiantes en foros de debate o mensajes privados con otros usuarios.

2.2. Web Semántica

La Web Semántica (Semantic Web, 2015) es una visión del futuro de la Web donde la información cuenta con un significado explícito, permitiendo que las máquinas puedan procesar automáticamente e integrar la información disponible en la Web.

Sus principios son crear una web orientada a los datos, que permita relacionar los datos entre sí, que se compartan, que se reutilicen y que puedan ser procesados de forma automática por las máquinas; todo esto nos lleva a poder inferir nuevas relaciones entre los datos que hasta ahora eran desconocidas.

Con el fin de conseguir lo descrito anteriormente, se necesita definir y describir las relaciones entre los datos que forman la web. En la Web Semántica, estas relaciones se definen usando los lenguajes RDF (Hayes & Patel-Schneider, 2014) y OWL (OWL 2 Web Ontology Language, 2012), ya que proporciona una definición formal para describir estas relaciones.

2.3. Ontologías

Las ontologías consisten en una representación formal del mundo real que define un conjunto de primitivas representativas con las cuales modelar un dominio de conocimiento (Ontologies, 2015). Estas primitivas son típicamente conceptos representados por clases, atributos representados por propiedades, miembros de clase representados por instancias de clase, y relaciones representadas por instancias de una propiedad.

2.3.1 OWL

OWL (del inglés Ontology Web Language) (McGuinness & Harmelen, 2004) es un lenguaje de descripción de ontologías que está pensado para ser usado cuando la información contenida en los documentos necesita ser procesada por las aplicaciones, al contrario que en las situaciones donde el contenido sólo necesita ser presentado a los humanos. OWL se usa para representar explícitamente el significado de términos en vocabularios y las relaciones entre esos términos. Esta representación de términos y sus interrelaciones se denomina ontología. OWL tiene mayor capacidad para expresar significado y semántica que XML, RDF y, de este modo, OWL va más allá de estos lenguajes en su capacidad para representar contenido interpretable por un ordenador en la Web.

A continuación, se describe la sintaxis de los elementos básicos necesarios para definir una ontología en OWL:

- Clases, proporcionan un mecanismo de abstracción para agrupar recursos con características similares. Cada clase OWL está asociada con un conjunto de individuos.

La sintaxis de definición de una clase se realiza entre '<>' especificando el prefijo owl:Class para indicar que se trata de una clase y a continuación la referencia a la URI de la clase en RDF.

```
<owl:Class rdf:about="http://ej.com/claseEjemplo"/>
```

- Propiedades de objeto, sirven para representar las relaciones entre instancias de dos clases.

La sintaxis de definición de una relación entre dos clases se realiza usando líneas delimitadas por '<>', el prefijo owl:ObjectProperty indica que se trata de una propiedad de objeto seguida de la referencia a URI de la propiedad que se quiere definir.

Para indicar el dominio de la relación se usa el prefijo rdfs:domain seguido de la URI de la clase perteneciente al dominio.

Para indicar el rango se usa el prefijo rdfs:range seguido de la URI de la clase perteneciente al rango.

```
<owl:ObjectProperty rdf:about="http://ej.com/nombrePropiedad">
  <rdfs:domain rdf:resource="http://ej.com/claseDominio"/>
  <rdfs:range rdf:resource="http://ej.com/claseRango"/>
</owl:ObjectProperty>
```

- Propiedades de datos, sirven para representar la relación de una clase con tipos de datos simples. OWL utiliza la mayoría de los tipos de datos definidos en el esquema XML. Las referencias a estos tipos de datos se hacen mediante la referencia a la URI para el tipo de dato concreto.

La sintaxis para declarar las propiedades de objetos se realizan mediante líneas delimitadas por '<>', el prefijo owl:DatatypeProperty indica que se trata de una propiedad de datos, a continuación se especifica la URI del nombre asignado a la propiedad.

Para indicar el dominio de la relación se usa el prefijo rdfs:domain seguido de la URI de la clase perteneciente al dominio.

Para indicar el rango se usa el prefijo rdfs:range, seguido de la URI del tipo primitivo definido en el esquema XML <http://www.w3.org/2001/XMLSchema#TipoDato>.

```
<owl:DatatypeProperty rdf:about="http://ej.com/nombrePropiedad">
  <rdfs:domain rdf:resource="http://ej.com/claseDominio"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#TipoDato"/>
</owl:DatatypeProperty>
```

Otra de las ventajas de OWL es que debido a su estructura jerárquica de clases, permite razonar e inferir información no declarada de forma automática (Casali & Gonzalez, 2018).

- Validación de la consistencia de una ontología: el razonador puede comprobar si una ontología no contiene hechos contradictorios
- Validación del cumplimiento de los conceptos de la ontología: el razonador determina si es posible que una clase tenga instancias.
- Clasificación de la ontología: el razonador computa a partir de los axiomas declarados, las relaciones de subclase entre todos los conceptos declarados explícitamente, a fin de construir la jerarquía de clases.
- Posibilita la resolución de consultas durante la recuperación de información basada en ontologías: a partir de la jerarquía de clases se pueden formular consultas como conocer todas las subclases de un concepto, inferir nuevas subclases de un concepto, las superclases directas, etc.
- Precisiones sobre los conceptos de la jerarquía: el razonador puede inferir cuáles son las clases a las que directamente pertenece y mediante la jerarquía inferida obtener todas las clases a las cuales indirectamente pertenece una clase o individuo dentro de la ontología.

2.4. RDF

RDF (del inglés Resource Description Framework) (Hayes & Patel-Schneider, 2014) es un modelo que proporciona un marco abstracto y conceptual capaz de definir y utilizar metadatos de recursos.

Para el mapeo de la base de datos relacional al esquema RDF se ha usado la sintaxis N-Tripletas. Las N-Tripletas son una secuencia de términos RDF que representan al sujeto, el predicado y el objeto; El predicado establece una direccionalidad desde el sujeto al objeto. El mismo recurso puede aparecer unas veces como sujeto y otras como objeto, lo cual permite encontrar conexiones entre diferentes tripletas. Las secuencias de términos pueden estar separadas por espacio en blanco o tabulador, la tripleta termina en un '.' y una línea nueva.

Los términos están formados por una IRI (Identificador de recursos Internacionalizado) que se definen entre '<' y '>'.

Ejemplo de sintaxis de una tripleta simple en RDF:

```
<http://ej.com/sujeto1> <http://ej.com/predicado1> <http://ej.com/objeto1> .
```

Ejemplo de sintaxis de literales en RDF:

```
<http://ej.com/sujeto1> <http://ej.com/propiedad1>  
"valor"^^<http://www.w3.org/2001/XMLSchema#Tipo> .
```

El objetivo general de RDF es poder fusionar automáticamente información de múltiples fuentes para formar una colección más grande que sea coherente y útil. Como punto de partida para esta fusión, toda la información se transmite siguiendo el esquema sujeto-predicado-objeto descrito anteriormente.

2.5. SPARQL

SPARQL (del inglés SPARQL RDF Query Language) (SPARQL 1.1 Overview, 2013) es un lenguaje que permite realizar consultas sobre RDF, y es capaz de extraer y manipular información almacenada en formato RDF. Tiene la capacidad de expresar consultas que permiten interrogar diferentes fuentes de datos. El resultado de estas consultas pueden ser un conjunto de resultados o un RDF.

El lenguaje soporta los operadores y cláusulas típicas de otros lenguajes de consulta como SQL tales como SELECT, COUNT, FILTER, GROUP BY, LIMIT, DESCRIBE, DISTINCT, etc. Aparte incluye otras cláusulas como ASK que devuelve un booleano si existe algún resultado coincidente con el patrón especificado. Otra peculiaridad es la necesidad de especificar la cláusula OPTIONAL para que se muestren resultados donde algún campo sea nulo.

Ejemplo de consulta en SPARQL que devuelve el número de visualizaciones de un estudiante para una asignatura determinada:

```
SELECT (COUNT(*) AS ?count_viewed) ?userid ?courseid
WHERE
{ ?x <http://khaos.uma.es/ontologies/moodle/logaction> ?action. FILTER (?action = 'viewed')
  ?x <http://khaos.uma.es/ontologies/moodle/logedulevel> ?edulevel. FILTER (?edulevel = 2)
  ?x <http://khaos.uma.es/ontologies/moodle/recordUser> ?userid.
  ?x <http://khaos.uma.es/ontologies/moodle/recordCourse> ?courseid.
}GROUP BY ?action ?edulevel ?userid ?courseid
```

Los resultados de las consultas en SPARQL son mapeados de variables a términos RDF. A menudo, la salida se muestra en forma tabular. A modo de ejemplo, en la Tabla 1 se muestran los resultados de la consulta anterior.

count_viewed	userid	courseid
464	http://khaos.uma.es/ontologies/moodle/user14261	http://khaos.uma.es/ontologies/moodle/course49
203	http://khaos.uma.es/ontologies/moodle/user5216	http://khaos.uma.es/ontologies/moodle/course36
38	http://khaos.uma.es/ontologies/moodle/user6285	http://khaos.uma.es/ontologies/moodle/course79

Tabla 1: Resultados consulta SPARQL.

2.6. Stardog

Stardog (Stardog, 2019) es un sistema gestor de base de datos semánticos que soporta los estándares RDF y OWL, y que permite evaluar consultas en el lenguaje SPARQL usando el endpoint del razonador de Stardog.

Stardog incorpora un repositorio donde se almacena la información en forma de tripletas en formato RDF, que están formadas como se ha detallado con anterioridad en el apartado 2.4. RDF

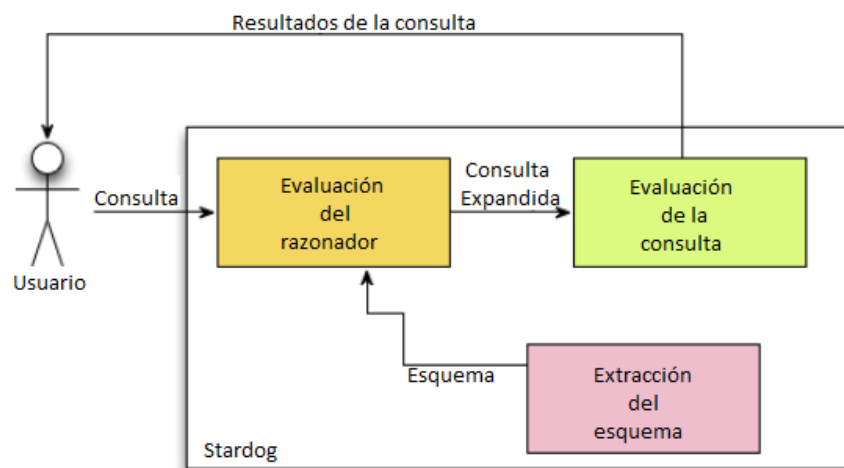


Figura 3: Flujo de ejecución de una consulta en Stardog.

La Figura 3 muestra el flujo de ejecución de una consulta en Stardog. El razonamiento en Stardog se basa en una técnica de reescritura de consultas: Stardog reescribe la consulta del usuario con respecto a cualquier esquema o regla definida en la ontología, y luego ejecuta la consulta expandida resultante contra los datos almacenados. Este proceso es completamente automático y no requiere intervención del usuario.

2.7. Moodle

Es un sistema gratuito para la implementación de los procesos de aprendizaje en línea. Permite a los docentes la creación de sus propios espacios para sus asignaturas, llenos de recursos dinámicos que extienden el aprendizaje, permitiendo el acceso al alumnado en cualquier momento y desde cualquier en cualquier sitio que disponga de conexión a internet.

Moodle es ampliamente usado a nivel mundial por las facilidades que ofrece a nivel organizativo, pedagógico y tecnológico. Cuenta con innumerables módulos predefinidos que facilitan una rápida implantación del sistema en cualquier comunidad educativa.

Arquitectura de Moodle

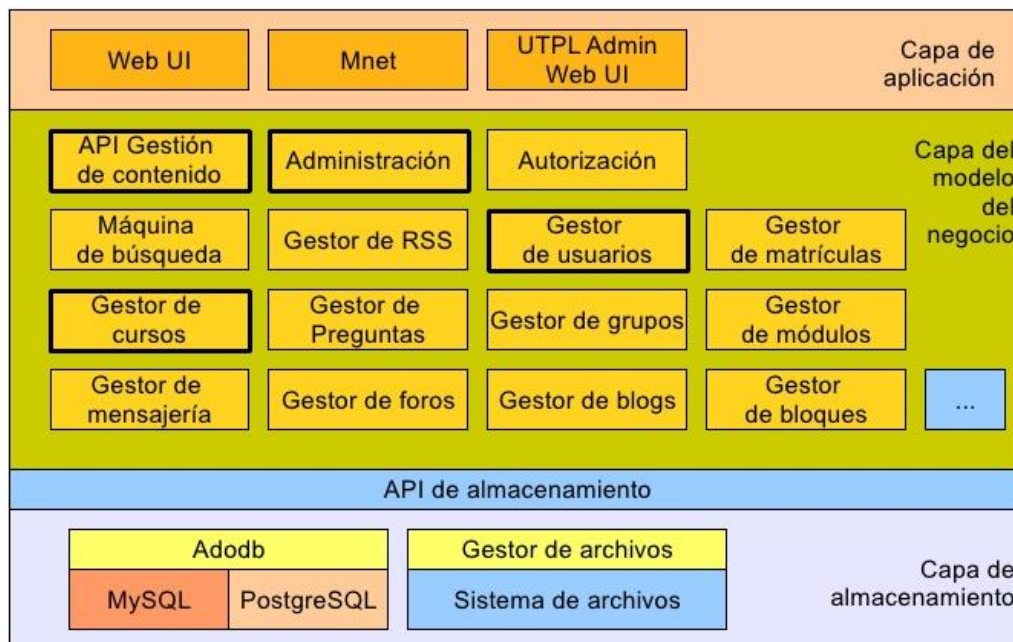


Figura 4: Arquitectura módulos principales de Moodle. Figura obtenida de (Granda, 2008)

La Figura 4 muestra la arquitectura de Moodle la cual está diseñada siguiendo el modelo de desarrollo basado en 3 capas. A continuación, se detalla brevemente cada capa:

- **Capa de aplicación:** Es la encargada de la presentación y a través de ella el usuario interactúa con el sistema.
- **Capa del modelo de negocio:** Esta capa es la que concentra toda la lógica del negocio. Está dividida en diversos módulos que implementan las diferentes funcionalidades de la plataforma y se encarga de comunicar la interfaz de usuario con la capa de datos.
- **Capa de datos:** Esta capa está formada por un gestor de base de datos y es la encargada de almacenar los datos y devolverlos a la capa de negocio cuando sean solicitados. Para este proyecto, esta es la capa más relevante, ya que los datos que se han facilitado desde el servicio de Enseñanza Virtual de la Universidad dentro del proyecto PIE17-166 son algunas tablas de la base de datos MySQL del Moodle de la Escuela Técnica Superior de Ingeniería Informática de la UMA.

Se ha estudiado la composición y relaciones de las tablas donde Moodle almacena la información. Cabe destacar la tabla 'logstore_standar_log' que podemos observar en la Figura 5 que contiene la información de todas las interacciones que realizan los usuarios en la plataforma, ya que es la que aporta más información del comportamiento de los usuarios. También se ha estudiado la información de las entregas con las tablas 'assign' y 'assign_submission' que podemos observar en la Figura 6, así como información de los usuarios que forman parte de una asignatura con el rol que tienen en dicha asignatura presente en la Figura 7.

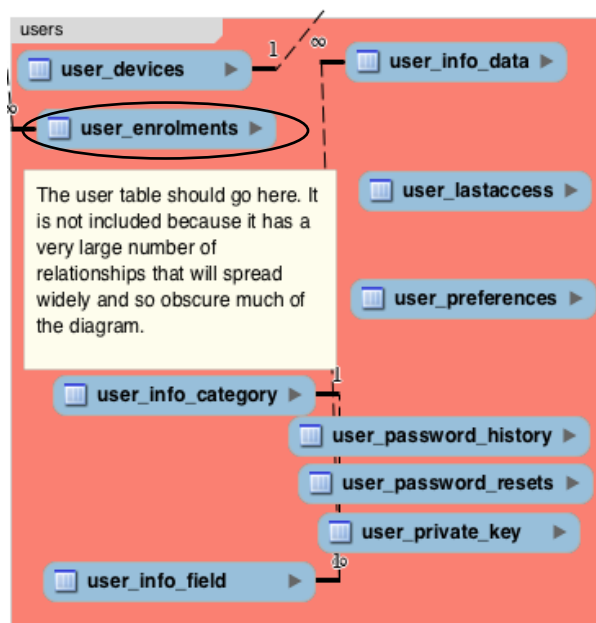


Figura 7: Modelo relacional de la parte de usuarios de Moodle.

3

Repositorio RDF

En este capítulo vamos a describir la ontología diseñada, cómo se han extraído los datos del sistema gestor de base de datos relacional, el alineamiento de los resultados de estas consultas a RDF y por último, el diseño de las consultas necesarias en SPARQL para realizar la fase de análisis.

3.1. Diseño de la ontología

La ontología propuesta, llamada 'moodle.owl', cuenta con un total de 6 clases (grupos de individuos con los mismos atributos), 9 propiedades de objeto (representan relaciones binarias entre individuos), 51 propiedades de datos (atributos individuales) y 120 axiomas de restricciones. La Figura 8 muestra una visión general de dicha ontología.

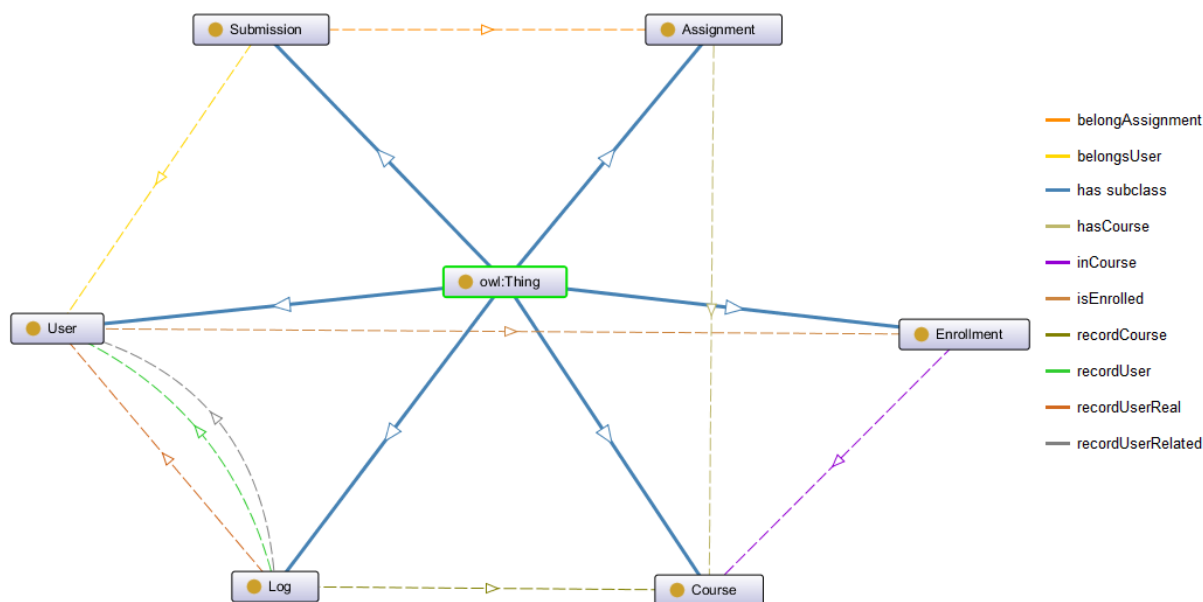


Figura 8: Visión general de la ontología Moodle.

La Tabla 2 muestra la propiedad y la descripción lógica de la clase `Course` que representa al conjunto de asignaturas que están presentes en el campus virtual. En este caso la única propiedad que está presente es el `id` que asigna el campus virtual al crear cada asignatura.

Propiedades de objeto	Descripción lógica
courseid	$\exists \text{ courseid } \text{DatatypeLiteral} \sqsubseteq \text{Course}$ $\top \sqsubseteq \forall \text{ courseid } \text{Datatypeint}$

Tabla 2: Clase curso, propiedades de objeto y de datos.

La Tabla 3 muestra las propiedades y la descripción lógica de la clase User, que representa al conjunto de usuarios que forman parte del campus virtual. Esta clase cuenta con dos propiedades: userid que es el identificador que se asigna a cada usuario en el momento en que se registra en la plataforma, y isEnrolled que se ha creado para representar que un usuario forma parte de una asignatura en el campus virtual.

Propiedades de objeto	Descripción lógica
isEnrolled	$\exists \text{ isEnrolled } \text{Thing} \sqsubseteq \text{User}$ $\top \sqsubseteq \forall \text{ isEnrolled } \text{Enrollment}$
userid	$\exists \text{ userid } \text{DatatypeLiteral} \sqsubseteq \text{User}$ $\top \sqsubseteq \forall \text{ userid } \text{Datatypeint}$

Tabla 3: Clase usuario, propiedades de objeto y de datos.

La Tabla 4 muestra las propiedades y la descripción lógica de la clase Enrollment, que representa el rol de un usuario en una asignatura. Esta clase cuenta con dos propiedades: role que indica el rol que tiene un usuario en una asignatura y cuyos valores más comunes son profesor o estudiante, y inCourse, que sirve para enlazar al usuario con la asignatura.

Propiedades de objeto	Descripción lógica
inCourse	$\exists \text{ inCourse } \text{Thing} \sqsubseteq \text{Enrollment}$ $\top \sqsubseteq \forall \text{ inCourse } \text{Course}$
role	$\exists \text{ role } \text{DatatypeLiteral} \sqsubseteq \text{Enrollment}$ $\top \sqsubseteq \forall \text{ role } \text{Datatypestring}$

Tabla 4: Clase matrícula, propiedades de objeto y de datos.

La Tabla 5 muestra las propiedades de la clase Log, que representa el registro de los eventos que realizan los usuarios. Esta clase cuenta con 19 propiedades, entre las que destacan para este proyecto: logedulevel que indica a qué tipo de usuario pertenece el evento, por ejemplo si el evento lo ha motivado un profesor el campo contiene el valor 1 y en el caso de que sea un estudiante contiene el valor 2, logaction indica qué tipo de acción ha realizado el usuario (los valores más comunes para esta propiedad son visualización y entrega), y logtimecreated que indica la marca temporal en la que se registró el evento.

Propiedades de objeto	Descripción lógica
recordCourse	$\exists \text{ recordCourse Thing} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ recordCourse Course}$
recordUser	$\exists \text{ recordUser Thing} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ recordUser User}$
recordUserReal	$\exists \text{ recordUserReal Thing} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ recordUserReal User}$
recordUserRelated	$\exists \text{ recordUserRelated Thing} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ recordUserRelated User}$
logaction	$\exists \text{ logaction DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ logaction Datatypestring}$
loganonymous	$\exists \text{ loganonymous DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ loganonymous Datatypeint}$
logcomponent	$\exists \text{ logcomponent DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ logcomponent Datatypestring}$
logcontextid	$\exists \text{ logcontextid DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ logcontextid Datatypeint}$
logcontextinstanceid	$\exists \text{ logcontextinstanceid DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ logcontextinstanceid Datatypeint}$
logcontextlevel	$\exists \text{ logcontextlevel DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ logcontextlevel Datatypeint}$
logcrud	$\exists \text{ logcrud DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ logcrud Datatypestring}$
logedulevel	$\exists \text{ logedulevel DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ logedulevel Datatypeint}$
logeventname	$\exists \text{ logeventname DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ logeventname Datatypestring}$
logid	$\exists \text{ logid DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ logid Datatypeint}$
logobjectid	$\exists \text{ logobjectid DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ logobjectid Datatypeint}$
logobjecttable	$\exists \text{ logobjecttable DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ logobjecttable Datatypestring}$
logorigin	$\exists \text{ logorigin DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ logorigin Datatypestring}$
logtarget	$\exists \text{ logtarget DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ logtarget Datatypestring}$
logtimecreated	$\exists \text{ logtimecreated DatatypeLiteral} \sqsubseteq \text{Log}$ $\top \sqsubseteq \forall \text{ logtimecreated DatatypedateTime}$

Tabla 5: Clase registro, propiedades de objeto y de datos.

La Tabla 6 muestra las propiedades de la clase ‘Assignment’, que representa las tareas que proponen los profesores. Esta clase cuenta con 27 propiedades que contienen la configuración de la tarea, por ejemplo: el número máximo de intentos (assignmentmaxattempts), fecha de entrega (assignmentduedate), la descripción de la tarea (assignmentintro), etc.

Propiedades de objeto	Descripción lógica
hasCourse	$\exists \text{ hasCourse Thing} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ hasCourse Course}$
assignmentallows submissionsfromdate	$\exists \text{ assignmentallows submissionsfromdate DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentallows submissionsfromdate DatatypeDateTime}$
assignmentalways showdescription	$\exists \text{ assignmentalways showdescription DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentalways showdescription Datatypeint}$
assignmentattemptreopenmethod	$\exists \text{ assignmentattemptreopenmethod DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentattemptreopenmethod Datatypestring}$
assignmentblindmarking	$\exists \text{ assignmentblindmarking DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentblindmarking Datatypeint}$
assignmentcompletionssubmit	$\exists \text{ assignmentcompletionssubmit DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentcompletionssubmit Datatypeint}$
assignmentcutoffdate	$\exists \text{ assignmentcutoffdate DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentcutoffdate DatatypeDateTime}$
assignmentduedate	$\exists \text{ assignmentduedate DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentduedate DatatypeDateTime}$
assignmentgrade	$\exists \text{ assignmentgrade DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentgrade Datatypeint}$
assignmentid	$\exists \text{ assignmentid DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentid Datatypeint}$
assignmentintro	$\exists \text{ assignmentintro DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentintro Datatypestring}$
assignmentintroformat	$\exists \text{ assignmentintroformat DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentintroformat Datatypeint}$
assignmentmarkingallocation	$\exists \text{ assignmentmarkingallocation DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentmarkingallocation Datatypeint}$
assignmentmarkingworkflow	$\exists \text{ assignmentmarkingworkflow DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentmarkingworkflow Datatypeint}$
assignmentmaxattempts	$\exists \text{ assignmentmaxattempts DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentmaxattempts Datatypeint}$
assignmentname	$\exists \text{ assignmentname DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentname Datatypestring}$
assignmentnosubmissions	$\exists \text{ assignmentnosubmissions DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentnosubmissions Datatypeint}$
assignmentrequireallteam memberssubmit	$\exists \text{ assignmentrequireallteam memberssubmit DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentrequireallteam memberssubmit Datatypeint}$
assignmentrequiresubmissionstatement	$\exists \text{ assignmentrequiresubmissionstatement DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentrequiresubmissionstatement Datatypeint}$
assignmentrevealidentities	$\exists \text{ assignmentrevealidentities DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentrevealidentities Datatypeint}$
assignmentsendlatenotifications	$\exists \text{ assignmentsendlatenotifications DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentsendlatenotifications Datatypeint}$
assignmentsendnotifications	$\exists \text{ assignmentsendnotifications DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentsendnotifications Datatypeint}$
assignmentsendstudentnotifications	$\exists \text{ assignmentsendstudentnotifications DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentsendstudentnotifications Datatypeint}$
assignmentsubmissionsdrafts	$\exists \text{ assignmentsubmissionsdrafts DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentsubmissionsdrafts Datatypeint}$
assignmentteams submission	$\exists \text{ assignmentteams submission DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentteams submission Datatypeint}$
assignmentteams submissionsgroupingid	$\exists \text{ assignmentteams submissionsgroupingid DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmentteams submissionsgroupingid Datatypeint}$
assignmenttimemodified	$\exists \text{ assignmenttimemodified DatatypeLiteral} \sqsubseteq \text{Assignment}$ $\top \sqsubseteq \forall \text{ assignmenttimemodified DatatypeDateTime}$

Tabla 6: Clase tarea, propiedades de objeto y de datos.

La Tabla 7 muestra las propiedades de la clase ‘Submission’, que representa las entregas que realizan los usuarios. Esta clase cuenta con 9 propiedades que almacenan la información de cada entrega, por ejemplo: ‘belongAssignment’ indica la tarea en la que se realiza la entrega, ‘belongsUser’ indica el usuario que realiza la entrega, ‘submissionattempnumber’ indica el número de intento de la entrega, ‘submissiongroupid’ es una propiedad opcional que indica el id del grupo en el caso de que se trate de una tarea grupal, ‘submissionid’ es el identificador que le asigna a la entrega la plataforma, ‘submissionlatest’ indica si se trata del último intento de entrega de ese usuario en esa tarea, ‘submissionstatus’ indica el estado de la entrega (borrador o entregado), ‘submissiontimecreated’ indica la marca de tiempo de la primera entrega en esta tarea y ‘submissiontimemodified’ indica la marca de tiempo de la modificación de la entrega.

Propiedades de objeto	Descripción lógica
belongAssignment	$\exists \text{ belongAssignment Thing} \sqsubseteq \text{Submission}$ $\top \sqsubseteq \forall \text{ belongAssignment Assignment}$
belongsUser	$\exists \text{ belongsUser Thing} \sqsubseteq \text{Submission}$ $\top \sqsubseteq \forall \text{ belongsUser User}$
submissionattempnumber	$\exists \text{ submissionattempnumber DatatypeLiteral} \sqsubseteq \text{Submission}$ $\top \sqsubseteq \forall \text{ submissionattempnumber Datatypeint}$
submissiongroupid	$\exists \text{ submissiongroupid DatatypeLiteral} \sqsubseteq \text{Submission}$ $\top \sqsubseteq \forall \text{ submissiongroupid Datatypeint}$
submissionid	$\exists \text{ submissionid DatatypeLiteral} \sqsubseteq \text{Submission}$ $\top \sqsubseteq \forall \text{ submissionid Datatypeint}$
submissionlatest	$\exists \text{ submissionlatest DatatypeLiteral} \sqsubseteq \text{Submission}$ $\top \sqsubseteq \forall \text{ submissionlatest Datatypeint}$
submissionstatus	$\exists \text{ submissionstatus DatatypeLiteral} \sqsubseteq \text{Submission}$ $\top \sqsubseteq \forall \text{ submissionstatus Datatypestring}$
submissiontimecreated	$\exists \text{ submissiontimecreated DatatypeLiteral} \sqsubseteq \text{Submission}$ $\top \sqsubseteq \forall \text{ submissiontimecreated DatatypedateTime}$
submissiontimemodified	$\exists \text{ submissiontimemodified DatatypeLiteral} \sqsubseteq \text{Submission}$ $\top \sqsubseteq \forall \text{ submissiontimemodified DatatypedateTime}$

Tabla 7: Clase entrega, propiedades de objeto y de datos.

3.2. Extracción de los datos del SGBD relacional

En la Figura 9 se muestra el modelo relacional de la copia de los datos facilitados para este proyecto. Consta de 4 tablas: '_logstore_standar_log', '_users_enrolled', '_assign' y '_assign_submission'.

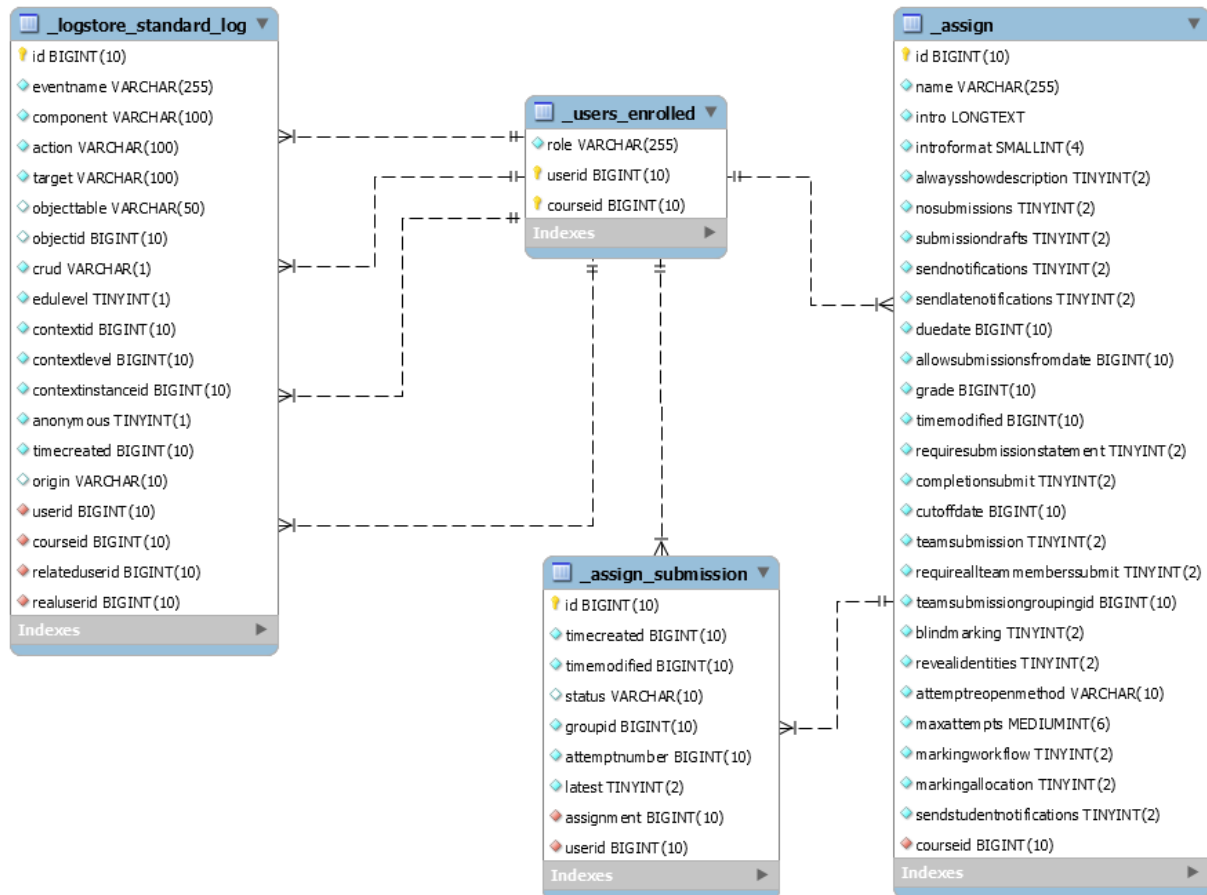


Figura 9: Modelo relacional de las tablas facilitadas por enseñanza virtual.

Para poblar la clase usuario y curso de individuos, se han usado las siguientes consultas SQL. Se aplica la cláusula DISTINCT para asegurar que los individuos son únicos.

```
SELECT DISTINCT userid FROM _users_enrolled;
```

```
SELECT DISTINCT courseid FROM _users_enrolled;
```

Una vez que se tiene la información de los cursos y de los usuarios, el siguiente paso sería obtener todos los datos de la tabla '_users_enrolled' para poder asignar el rol que tiene cada individuo en una asignatura concreta. Esto se realiza con la siguiente consulta:

```
SELECT * FROM _users_enrolled;
```

Para poblar la clase registro de individuos, se ha usado la siguiente consulta. Se crea un campo con el campo de fecha transformado, ya que el formato XML para fecha y hora lleva el carácter T para separar la fecha de la hora en lugar de un espacio.

```
SELECT *,replace(FROM_UNIXTIME(timecreated), ' ', 'T') AS timecreated_transform
FROM _logstore_standard_log;
```

Para poblar la clase tarea de individuos, se ha usado la siguiente consulta. Se crean campos con las fechas transformadas por el motivo comentado anteriormente, y en el campo 'intro' se eliminan los caracteres de retorno de carro y salto de línea que contiene este campo, para dejarlo en texto de una sola línea.

```
SELECT *,
replace(FROM_UNIXTIME(duedate), ' ', 'T') AS duedate_transform,
replace(FROM_UNIXTIME(allowsubmissionsfromdate), ' ', 'T') AS allowsubmissionsfromdate_transform,
replace(FROM_UNIXTIME(timemodified), ' ', 'T') AS timemodified_transform,
replace(FROM_UNIXTIME(cutoffdate), ' ', 'T') AS cutoffdate_transform,
replace(replace(replace(intro, '\n', ' '), '\r', ' '), '\", '\"') AS intro_transform,
replace(name, '\", '\"') AS name_transform
FROM _assign;
```

Finalmente, solo queda poblar de individuos la clase entrega. Para ello se ha usado la siguiente consulta. Los únicos campos que hay que transformar son las fechas como se ha descrito anteriormente.

```
SELECT *,
replace(FROM_UNIXTIME(timecreated), ' ', 'T') AS timecreated_transform,
replace(FROM_UNIXTIME(timemodified), ' ', 'T') AS timemodified_transform
FROM _assign_submission;
```

3.3. Alineamiento de los resultados de las consultas SQL a RDF

Tras obtener los resultados de las consultas SQL que se han detallado en el apartado anterior, se establece el alineamiento de las propiedades de las clases de la ontología con las columnas que devuelven las consultas. En la Tabla 8 se detalla el alineamiento de las propiedades de datos de la clase tarea, en la Tabla 9 se detalla el alineamiento de las propiedades de datos la clase entrega, en la Tabla 10 se detalla el alineamiento de las propiedades de datos de la clase registro, en la Tabla 11 se detalla el alineamiento de las

propiedades de datos de la clase usuario, en la Tabla 12 se detalla el alineamiento de las propiedades de datos de la clase curso. Por último, en la Tabla 13 se detalla el alineamiento de las propiedades de datos de la clase matrícula.

Ha sido necesario crear la clase matrícula ya que, para almacenar el rol de un usuario en una asignatura, se necesita una herencia de la clase asignatura y de la clase usuario. Para representar esta información en la ontología se han usado la propiedad de objeto 'isEnrolled', cuyo valor para cada usuario es:

`http://khaos.uma.es/ontologies/moodle/enrolledu'id_usuario'c'id_curso'id'id_matricula'`

En el caso de las propiedades de objeto el alineamiento es el siguiente:

- belongAssignment se alinea con el individuo de la clase tarea en la que se ha realizado la entrega, identificándola por su IRI. Esta IRI se construye de la siguiente forma: `http://khaos.uma.es/ontologies/moodle/assignment'id_tarea'`
- belongsUser se alinea con el individuo de la clase usuario que realiza la entrega de la tarea, identificándolo por su IRI. Esta IRI se construye de la siguiente forma: `http://khaos.uma.es/ontologies/moodle/user'id_usuario'`
- hasCourse se alinea con el individuo de la clase asignatura a la cual pertenece la tarea, identificándola por su IRI. Esta IRI se construye de la siguiente forma: `http://khaos.uma.es/ontologies/moodle/course'id_asignatura'`
- inCourse se alinea con el individuo de la clase asignatura en la que un usuario está matriculado, identificándola por su IRI. Esta IRI se construye de la siguiente forma: `http://khaos.uma.es/ontologies/moodle/course'id_asignatura'`
- recordCourse se alinea con el individuo de la clase asignatura en la que se ha disparado un evento en el registro, identificándola con su IRI. Esta IRI se construye de la siguiente forma: `http://khaos.uma.es/ontologies/moodle/course'id_asignatura'`
- recordUser se alinea con el individuo que representa al usuario que ha disparado el evento en el registro, identificándolo por su IRI. Esta IRI se construye de la siguiente forma: `http://khaos.uma.es/ontologies/moodle/user'id_usuario'`
- recordUserReal es una propiedad opcional que se alinea con el individuo que representa al usuario cuando está identificado en el sistema y ha disparado un evento en el registro, identificándolo por su IRI. Esta IRI se construye de la siguiente forma: `http://khaos.uma.es/ontologies/moodle/user'id_usuario'`
- recordUserRelated es una propiedad opcional que se alinea con el individuo que representa al usuario sobre el que se han otorgado permisos, este hecho dispara un evento en el registro, al usuario se le identifica por su IRI. Esta IRI se construye de la siguiente forma: `http://khaos.uma.es/ontologies/moodle/user'id_usuario'`

Para realizar este proceso se han desarrollado 4 scripts en Python que ejecutan las consultas con un conector al sistema gestor de base de datos relacional y realizan la traducción de los resultados a la sintaxis de tripletas RDF, almacenando los resultados en ficheros de texto con la extensión '.nt', que es la que recomienda el consorcio W3C.

Propiedad de objeto	Columna en modelo relacional
assignmentallows submissionsfromdate	_assign.allows submissionsfromdate
assignmentalways showdescription	_assign.always showdescription
assignmentattemptreopenmethod	_assign.attemptreopenmethod
assignmentblindmarking	_assign.blindmarking
assignmentcompletions submit	_assign.completions submit
assignmentcutoffdate	_assign.cutoffdate
assignmentduedate	_assign.duedate
assignmentgrade	_assign.grade
assignmentid	_assign.id
assignmentintro	_assign.intro
assignmentintroformat	_assign.introformat
assignmentmarkingallocation	_assign.markingallocation
assignmentmarkingworkflow	_assign.markingworkflow
assignmentmaxattempts	_assign.maxattempts
assignmentname	_assign.name
assignmentnosubmissions	_assign.nosubmissions
assignmentrequireallteam members submit	_assign.requireallteam members submit
assignmentrequiresubmissionstatement	_assign.requiresubmissionstatement
assignmentrevealidentities	_assign.revealidentities
assignmentsendlatenotifications	_assign.sendlatenotifications
assignmentsendnotifications	_assign.sendnotifications
assignmentsendstudentnotifications	_assign.sendstudentnotifications
assignmentsubmissions drafts	_assign.submissions drafts
assignmentteams submission	_assign.teams submission
assignmentteams submissionsgroupingid	_assign.teams submissionsgroupingid
assignmenttimemodified	_assign.timemodified

Tabla 8: Alineamiento de las propiedades de la clase tarea.

Propiedad de objeto	Columna en modelo relacional
submissionattemptnumber	_assign_submission.attemptnumber
submissiongroupid	_assign_submission.groupid
submissionid	_assign_submission.id
submissionlatest	_assign_submission.latest
submissionstatus	_assign_submission.status
submissiontimecreated	_assign_submission.timecreated
submissiontimemodified	_assign_submission.timemodified

Tabla 9: Alineamiento de las propiedades de la clase entrega.

Propiedad de objeto	Columna en modelo relacional
logaction	_logstore_standard_log.action
loganonymous	_logstore_standard_log.anonymous
logcomponent	_logstore_standard_log.component
logcontextid	_logstore_standard_log.contextid
logcontextinstanceid	_logstore_standard_log.contextinstanceid
logcontextlevel	_logstore_standard_log.contextlevel
logcrud	_logstore_standard_log.crud
logedulevel	_logstore_standard_log.edulevel
logeventname	_logstore_standard_log.eventname
logid	_logstore_standard_log.id
logobjectid	_logstore_standard_log.objectid
logobjecttable	_logstore_standard_log.objecttable
logorigin	_logstore_standard_log.origin
logtarget	_logstore_standard_log.target
logtimecreated	_logstore_standard_log.timecreated

Tabla 10: Alineamiento de las propiedades de la clase registro.

Propiedad de objeto	Columna en modelo relacional
userid	_users_enrolled.userid

Tabla 11: Alineamiento de las propiedades de la clase usuario.

Propiedad de objeto	Columna en modelo relacional
courseid	_users_enrolled.courseid

Tabla 12: Alineamiento de las propiedades de la clase curso.

Propiedad de objeto	Columna en modelo relacional
role	_users_enrolled.role

Tabla 13: Alineamiento de las propiedades de la clase matrícula.

3.4. Diseño de las consultas en SPARQL necesarias

Para realizar el análisis de la actividad de los usuarios en el campus virtual, se han diseñado las consultas necesarias para obtener los datos. Con estos datos se realizarán estadísticas básicas y servirán para alimentar los algoritmos de machine learning.

La Consulta 1 devuelve el número de visualizaciones que ha realizado cada usuario en una asignatura. Para ello, en la cláusula WHERE se filtran los resultados para que la propiedad acción sea una visualización, y para limitarla a alumnos se filtra la propiedad edulevel al valor 2. Después, se aplica la cláusula GROUP BY y con la función COUNT se calcula el número de visualizaciones.

```
SELECT (COUNT(*) AS ?count_viewed) ?userid ?courseid
WHERE
{
  ?x <http://khaos.uma.es/ontologies/moodle/logaction> ?action. FILTER (?action = 'viewed')
  ?x <http://khaos.uma.es/ontologies/moodle/logedulevel> ?edulevel. FILTER (?edulevel = 2)
  ?x <http://khaos.uma.es/ontologies/moodle/recordUser> ?userid.
  ?x <http://khaos.uma.es/ontologies/moodle/recordCourse> ?courseid.
}GROUP BY ?action ?edulevel ?userid ?courseid
```

Consulta 1: Número de visualizaciones por usuario y asignatura.

La Consulta 2 devuelve el número de entregas que ha realizado cada usuario en una asignatura. Para ello, en la cláusula WHERE se filtran los resultados para que la propiedad acción sea una entrega y para limitarla a alumnos se filtra la propiedad edulevel al valor 2. Tras esto, se aplica la cláusula GROUP BY y con la función COUNT se calcula el número de entregas.

```
SELECT (COUNT(*) AS ?count_submission) ?userid ?courseid
WHERE
{
  ?x <http://khaos.uma.es/ontologies/moodle/logaction> ?action. FILTER (?action = 'submitted')
  ?x <http://khaos.uma.es/ontologies/moodle/logedulevel> ?edulevel. FILTER (?edulevel = 2)
  ?x <http://khaos.uma.es/ontologies/moodle/recordUser> ?userid.
  ?x <http://khaos.uma.es/ontologies/moodle/recordCourse> ?courseid.
}GROUP BY ?action ?edulevel ?userid ?courseid
```

Consulta 2: Número de entregas por usuario y asignatura.

La Consulta 3 devuelve los datos necesarios para representar una serie temporal con las visualizaciones de todos los usuarios. Para ello en la cláusula WHERE se filtran los resultados para que la propiedad acción sea una visualización y para limitarla a alumnos se filtra la propiedad edulevel al valor 2. Después se aplica la cláusula GROUP BY y con la función COUNT se calcula el número de visualizaciones.

```
SELECT (COUNT(*) AS ?count_viewed) ?timecreated
WHERE
{ ?x <http://khaos.uma.es/ontologies/moodle/logaction> ?action. FILTER (?action = 'viewed')
  ?x <http://khaos.uma.es/ontologies/moodle/logedulevel> ?edulevel. FILTER (?edulevel = 2)
  ?x <http://khaos.uma.es/ontologies/moodle/logtimecreated> ?timecreated.
}GROUP BY ?action ?edulevel ?timecreated
```

Consulta 3: Número de visualizaciones por fecha.

Consulta 4 devuelve los datos necesarios para representar una serie temporal con las visualizaciones de cada usuario en cada asignatura. Para ello en la cláusula WHERE se filtran los resultados para que la propiedad acción sea una visualización y para limitarla a alumnos se filtra la propiedad edulevel al valor 2. Tras esto se aplica la cláusula GROUP BY y con la función COUNT se calcula el número de visualizaciones.

```
SELECT (COUNT(*) AS ?count_viewed) ?userid ?courseid ?timecreated
WHERE
{ ?x <http://khaos.uma.es/ontologies/moodle/logaction> ?action. FILTER (?action = 'viewed')
  ?x <http://khaos.uma.es/ontologies/moodle/logedulevel> ?edulevel. FILTER (?edulevel = 2)
  ?x <http://khaos.uma.es/ontologies/moodle/recordUser> ?userid.
  ?x <http://khaos.uma.es/ontologies/moodle/recordCourse> ?courseid.
  ?x <http://khaos.uma.es/ontologies/moodle/logtimecreated> ?timecreated.
}GROUP BY ?action ?edulevel ?userid ?courseid ?timecreated
```

Consulta 4: Número de visualizaciones por usuario, asignatura y fecha.

La Consulta 5 devuelve los datos necesarios para representar una serie temporal con las entregas de todos los usuarios. Para ello en la cláusula WHERE se filtran los resultados para que la propiedad acción sea una entrega, para limitarla a alumnos se filtra la propiedad edulevel al valor 2. Tras esto se aplica la cláusula GROUP BY y con la función COUNT se calcula el número de entregas.

```
SELECT (COUNT(*) AS ?count_submitted) ?timecreated
WHERE
{ ?x <http://khaos.uma.es/ontologies/moodle/logaction> ?action. FILTER (?action = 'submitted')
  ?x <http://khaos.uma.es/ontologies/moodle/logedulevel> ?edulevel. FILTER (?edulevel = 2)
  ?x <http://khaos.uma.es/ontologies/moodle/logtimecreated> ?timecreated.
}GROUP BY ?action ?edulevel ?timecreated
```

Consulta 5: Número de entregas por fecha.

La Consulta 6 devuelve los datos necesarios para representar una serie temporal con las entregas de cada usuario en cada asignatura. Para ello en la cláusula WHERE se filtran los resultados para que la propiedad acción sea una visualización, para limitarla a alumnos se filtra la propiedad edulevel al valor 2. Tras esto se aplica la cláusula GROUP BY y con la función COUNT se calcula el número de visualizaciones.

```
SELECT (COUNT(*) AS ?count_submitted) ?userid ?courseid ?timecreated
WHERE
{ ?x <http://khaos.uma.es/ontologies/moodle/logaction> ?action. FILTER (?action = 'submitted')
  ?x <http://khaos.uma.es/ontologies/moodle/logedulevel> ?edulevel. FILTER (?edulevel = 2)
  ?x <http://khaos.uma.es/ontologies/moodle/recordUser> ?userid.
  ?x <http://khaos.uma.es/ontologies/moodle/recordCourse> ?courseid.
  ?x <http://khaos.uma.es/ontologies/moodle/logtimecreated> ?timecreated.
}GROUP BY ?action ?edulevel ?userid ?courseid ?timecreated
```

Consulta 6: Número de entregas por fecha, usuario, asignatura y fecha.

4

Análisis de datos

En este capítulo, vamos a abarcar la fase de análisis de los datos. Este análisis se basa en series temporales del número de visualizaciones y del número de tareas entregadas. También se va a hacer uso de algoritmos de aprendizaje máquina para agrupar a los alumnos en diferentes tipos en base a la interacción que realizan con el campus virtual. Para finalizar esta fase, se va a entrenar un modelo predictor que permite predecir el grupo al que pertenece un alumno a través de los datos de su interacción con la plataforma.

4.1. Análisis temporal

Como resultado de la Consulta 3 se obtiene la serie temporal agrupada por semanas. Con estos datos se genera la Figura 10, en la que se puede observar cómo en el periodo vacacional de Navidad descenden notablemente las visualizaciones, al igual que después de los exámenes de febrero, y también se pueden apreciar las festividades de la constitución en diciembre, la de todos los santos y el día de la hispanidad. En estos casos la bajada de visitas es menos pronunciada.

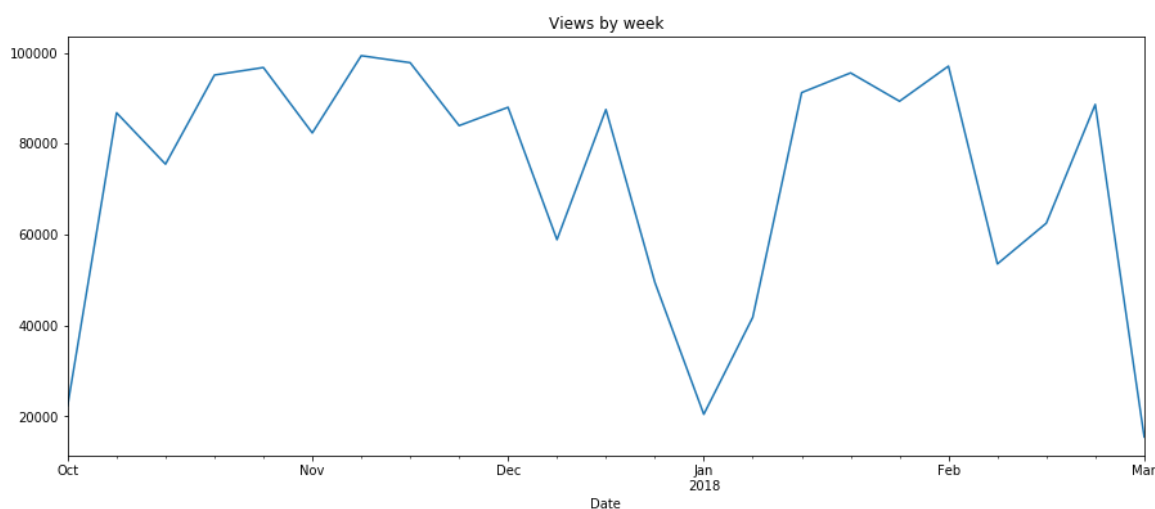


Figura 10: Serie temporal del número de visualizaciones.

En la Figura 11 se puede observar un ejemplo de uno de los resultados de la Consulta 4 agrupada por semanas, en la que se muestra la serie temporal con las visualizaciones del usuario 913 en la asignatura 129. Este es un usuario promedio, ya que sigue los patrones generales de actividad comentados en la Figura 10.

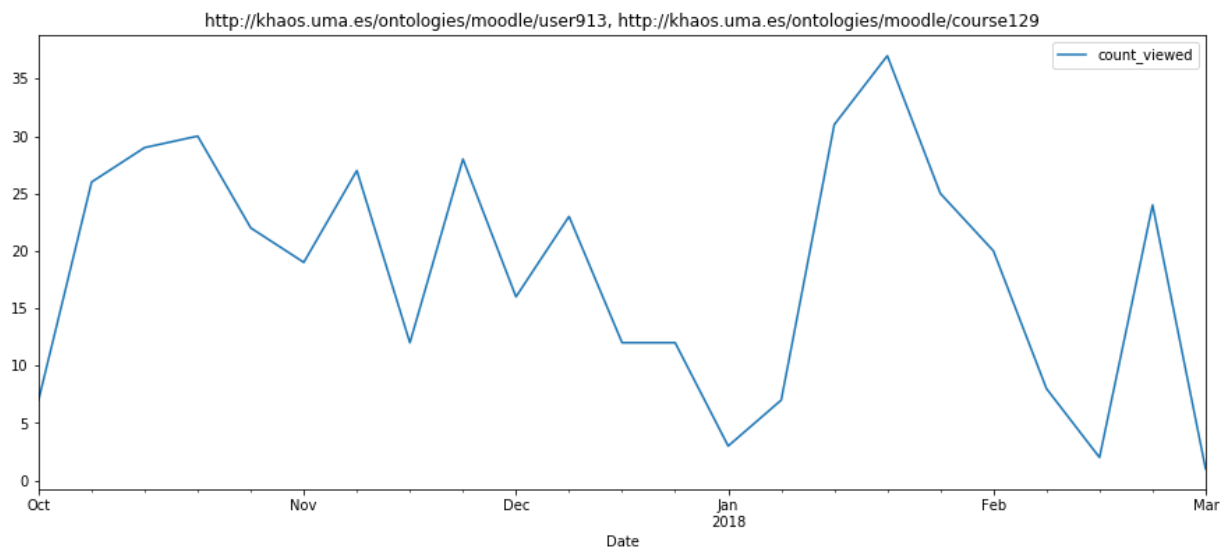


Figura 11: Serie temporal de las visualizaciones de un usuario en una asignatura.

Como resultado de la Consulta 5 se obtiene la serie temporal agrupada por semanas. Con estos datos se genera la Figura 12, en la que se puede observar cómo en el periodo vacacional de Navidad bajan notablemente las entregas. Se observa también, que a mediados de enero desciende fuertemente el número de entregas. Esto es totalmente normal, ya que en estas fechas es cuando se empiezan a preparar los exámenes finales. Cabe destacar que también se puede observar cómo el inicio de las entregas de tareas se produce más tarde que el de las visualizaciones. Esto es debido a que hasta que no se ha impartido temario suficiente, no se habilita la entrega de tareas.

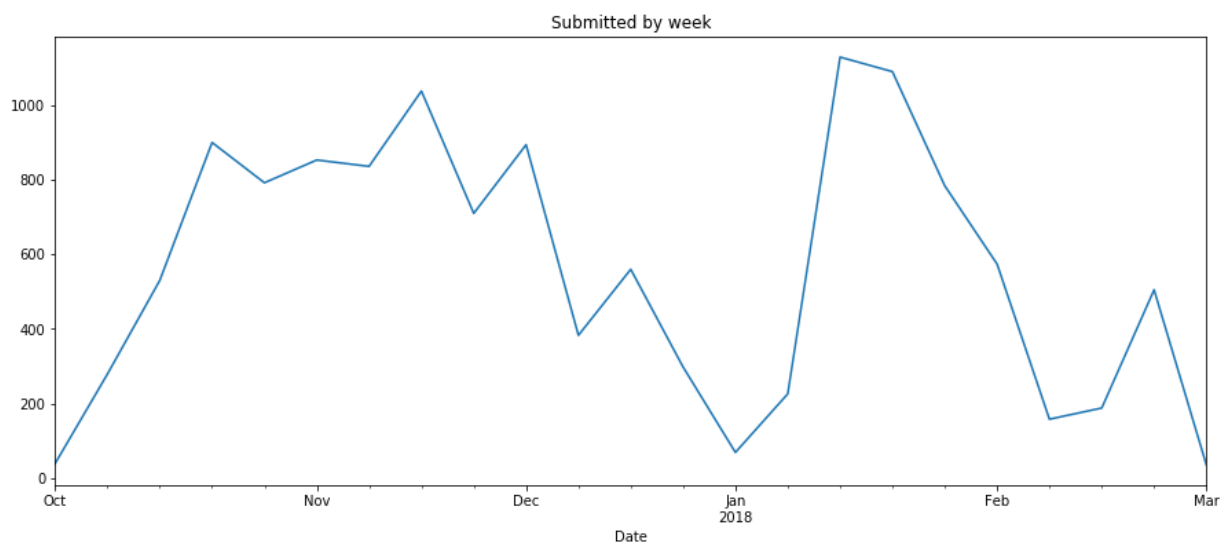


Figura 12: Serie temporal del número de entregas.

En la Figura 13 se puede observar un ejemplo de uno de los resultados de la Consulta 6, en la que se muestra la serie temporal con las visualizaciones del usuario 2265 en la asignatura 25. En el caso de este usuario se observa como desciende la entrega en los periodos vacacionales.

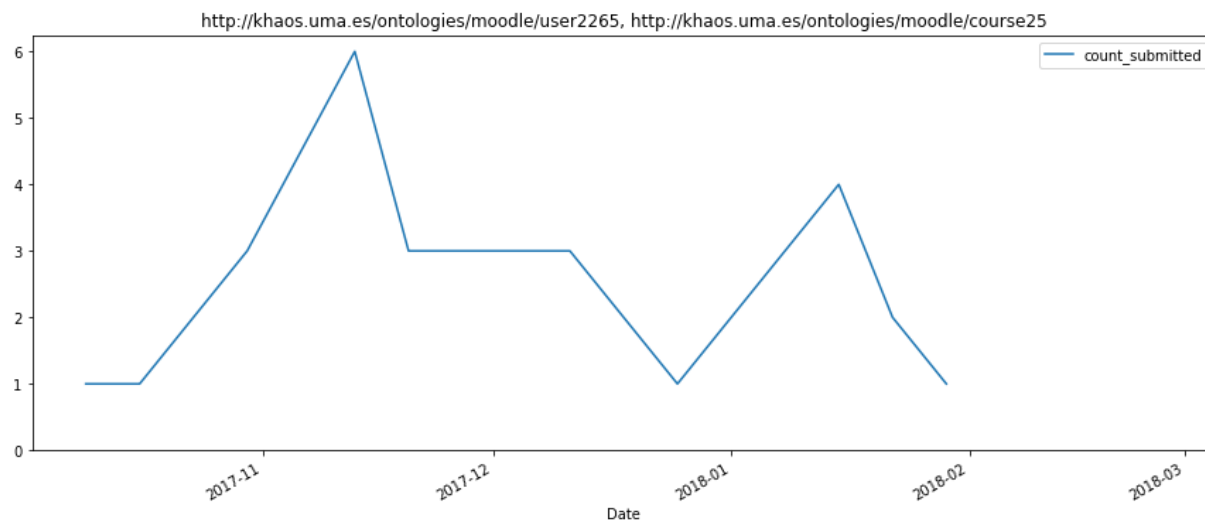


Figura 13: Serie temporal de las entregas de un usuario en una asignatura.

4.2. Análisis de grupos de usuarios

Como resultado de la unión de los resultados de la Consulta 1 y la Consulta 2, y tras estandarizar los datos para cada asignatura, se obtiene la representación del número de entregas y número de visualizaciones que realizan los alumnos de una asignatura. Cada punto de esta gráfica equivale al comportamiento de un alumno en una asignatura. Se puede observar en la Figura 14 un ejemplo del resultado para la asignatura 88.

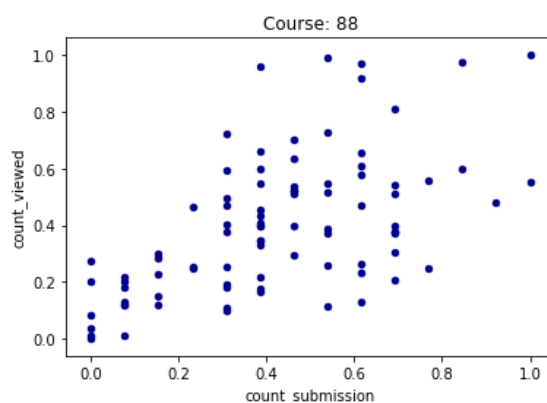


Figura 14: Número de entregas y visualizaciones en una asignatura.

Para preparar los datos que alimentan el algoritmo de agrupamiento, se obtiene los datos de visualizaciones y entregas de todas las asignaturas (ver Figura 15). Cada punto representa el comportamiento de un alumno en una asignatura.



Figura 15: Número de visualizaciones y entregas en todas las asignaturas.

Ahora que ya se tienen los datos preparados, se ejecuta el algoritmo de agrupamiento Kmeans de la librería scikit-learn que consiste en dividir un conjunto de muestras en grupos disjuntos. De forma abreviada, el algoritmo consta de tres pasos:

- El primer paso es la inicialización del algoritmo. Consiste en elegir los centroides iniciales, y el método más básico es elegir k muestras aleatorias del conjunto de datos.

Después de la inicialización, el algoritmo K-means consiste en hacer un bucle entre los dos pasos restantes.

- El segundo paso asigna cada muestra a su centroide más cercano.
- El tercer paso crea los nuevos centroides. Para ello calcula el valor medio de todas las muestras asignadas a cada centroide anterior y se calcula la diferencia entre el centroide antiguo y el nuevo. El algoritmo repite estos dos últimos pasos hasta que este valor sea inferior a un umbral. En otras palabras, se repite hasta que los centroides no se mueven significativamente.

Una vez que ha terminado la ejecución del algoritmo de agrupamiento KMeans, obtenemos la Figura 16, en la que se puede observar los 4 grupos de usuarios que ha establecido el algoritmo, junto con sus centroides en rojo que serían los representantes de ese grupo.

- El grupo 0 se caracteriza por realizar una entrega media de tareas y un número bajo de visualizaciones.
- El grupo 1 se caracteriza por un elevado número de visualizaciones.
- El grupo 2 se caracteriza por un número elevado de entregas.
- El grupo 3 se caracteriza por su bajo número de visualizaciones y un nivel bajo de entregas.

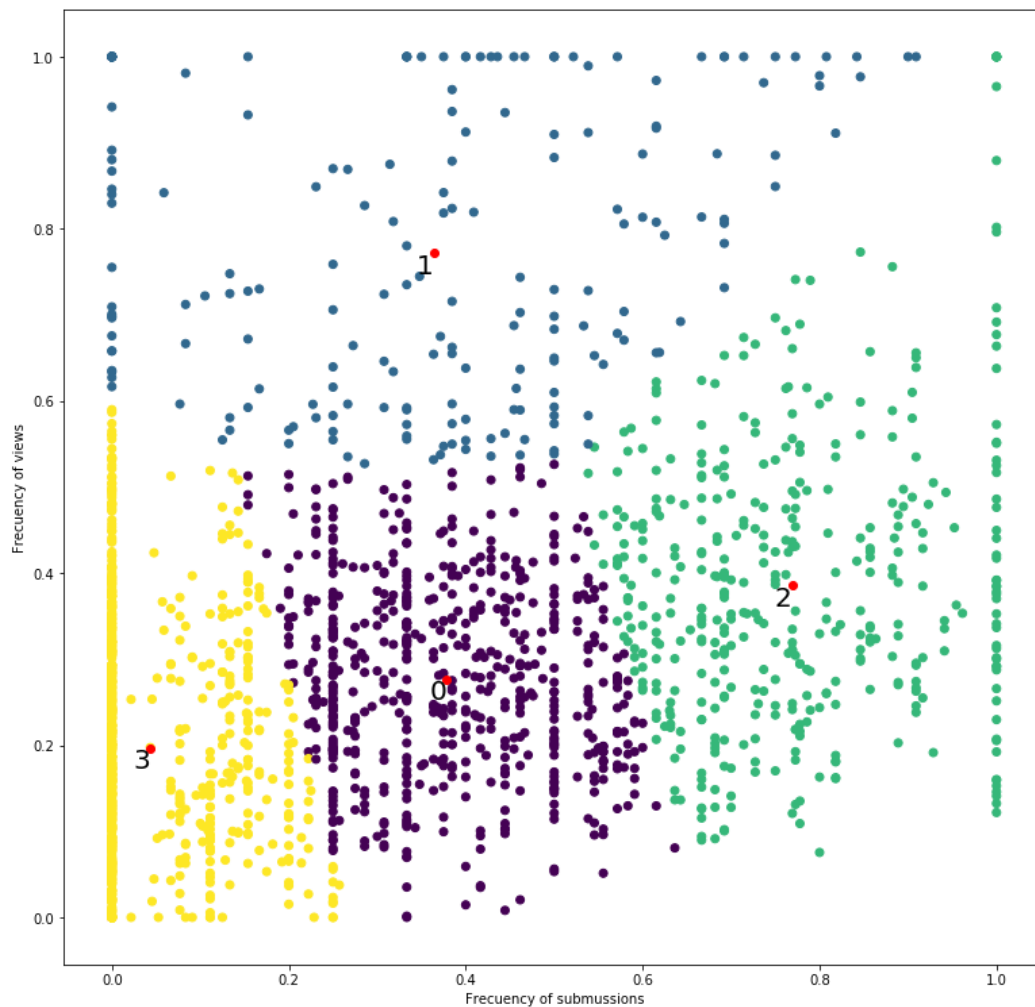


Figura 16: Grupos de usuarios definidos por el algoritmo k-means.

Ya que tenemos un conjunto de datos con sus características (número de visualizaciones y número de entregas) y una etiqueta con el grupo al que pertenecen, se va a entrenar un modelo predictor de machine learning.

Se ha elegido el modelo predictor KNeighborsClassifier de la librería scikit-learn. En su fase de entrenamiento, este modelo almacena los datos con los que se entrena. Para realizar la predicción, se basa en calcular la clase a la que pertenece el individuo a predecir. Para ello se realiza una votación por mayoría entre los vecinos más cercanos al punto. Finalmente, se le asigna la clase que tiene la mayor cantidad de representantes dentro de los vecinos más cercanos del punto.

Para realizar la fase de entrenamiento se ha usado un porcentaje del 70% de los datos etiquetados y se ha reservado un 30% para realizar la validación final de la calidad del modelo.

Se ha usado un ajuste de parámetros por rejilla para buscar la mejor configuración del modelo. Para ello, se han probado las diferentes métricas preestablecidas para el cálculo de la distancia, y los diferentes tipos de pesos que se le pueden asignar a los vecinos a la hora de emitir su voto. El último parámetro que se ha ajustado es p , que indica la potencia a usar en el caso particular de la métrica de Minkowski.

Los parámetros de la mejor configuración del modelo son:

- Métrica: la distancia euclídea.
- Número de grupos: 4
- Pesos de los votos de los vecinos: se ha elegido distancia, ya que hace que los vecinos más cercanos tengan un mayor peso en la votación.

La precisión obtenida en el conjunto de validación es de un 99.1776% de alumnos bien clasificados.

En la Figura 17 se puede observar la matriz de confusión que muestra el porcentaje de acierto para cada uno de los grupos, y permite saber a qué clases se están asignando los alumnos incorrectamente predichos. Se puede afirmar que el modelo tiene un buen comportamiento para todas las clases de alumnos.

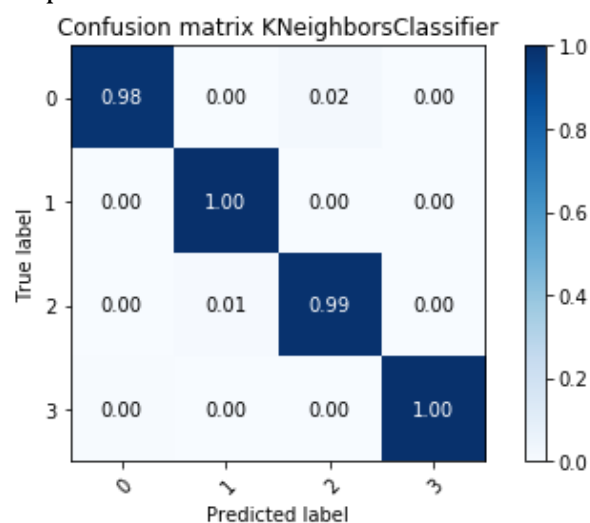


Figura 17: Matriz de confusión del modelo.

4.3. API

En esta sección, vamos a abarcar el desarrollo de una API para facilitar la integración con futuras aplicaciones que necesiten obtener los datos del análisis realizado.

Para implementar esta API se ha hecho uso de la librería Pyramid (The Pyramid Web Framework, 2019), esta librería permite desplegar un servidor de forma rápida y sencilla. El control del flujo de ejecución se realiza especificando rutas con anotaciones para cada una de las posibles llamadas que se quieran definir.

Se han definido dos llamadas de tipo GET que devuelven los datos más interesantes para el personal docente, respecto al rendimiento en su asignatura. Concretamente estas llamadas devuelven los datos del análisis para un usuario en una asignatura y otra que devuelve los datos de todos los usuarios de una asignatura.

- Llamada: `/statistics_user_course?user_id=X&course_id=Y`, esta llamada devuelve un JSON, del cual se puede observar un ejemplo en la Figura 19. Este JSON contiene el número de entregas y visualizaciones normalizado en el intervalo [0-1] junto con el clúster al que pertenece el usuario en esa asignatura según el algoritmo predictor entrenado en el capítulo 4.2. **Análisis de grupos de usuarios**
Aparte devuelve las series temporales por semanas con el número de visualizaciones y entregas para ese usuario en esa asignatura, estas series son las mostradas en el capítulo 4.1. **Análisis temporal.**
- Llamada: `/statistics_course?course_id=X`, esta llamada devuelve un JSON con la información de todos los alumnos de una asignatura. Se puede observar un ejemplo de la estructura del JSON en la Figura 18. Este JSON contiene la lista de alumnos, el número de entregas y visualizaciones normalizado en el intervalo [0-1] junto con el clúster al que pertenece cada usuario en esa asignatura según el algoritmo predictor entrenado en el capítulo 4.2. **Análisis de grupos de usuarios.**

```

{
  "cluster": {
    "class": 3,
    "%submissions": 0,
    "%views": 0
  },
  "views_series": {
    "timeStamps": {
      "0": 1508716800000000000,
      "1": 1509321600000000000,
      "2": 1509926400000000000,
      "3": 1510531200000000000,
      "4": 1511136000000000000,
      "5": 1511740800000000000,
      "6": 1512345600000000000,
      "7": 1512950400000000000
    },
    "views": {
      "0": 7,
      "1": 6,
      "2": 0,
      "3": 3,
      "4": 1,
      "5": 22,
      "6": 17,
      "7": 10
    }
  },
  "submissions_series": {
    "timeStamps": {
      "0": 1512345600000000000
    },
    "submissions": {
      "0": 1
    }
  }
}

```

Figura 19: Ejemplo del resultado de la llamada 'statistics_user_course'.

```

{
  "course_52": {
    "userid": {
      "0": 3932,
      "1": 1064,
      "2": 11273,
      "3": 7791,
      "4": 1628
    },
    "class": {
      "0": 3,
      "1": 3,
      "2": 2,
      "3": 3,
      "4": 3
    },
    "%submission": {
      "0": 0,
      "1": 0,
      "2": 1,
      "3": 0,
      "4": 0
    },
    "%viewed": {
      "0": 0,
      "1": 0.10526315789473684,
      "2": 0.9999999999999999,
      "3": 0.13157894736842102,
      "4": 0.42105263157894735
    }
  }
}

```

Figura 18: Ejemplo del resultado de la llamada 'statistics_course'.

5

Conclusiones

Como resultado de este trabajo, se propone un enfoque semántico que utiliza una ontología como base formal para representar toda la información que registra el campus virtual de la Escuela Técnica Superior de Ingeniería Informática relativa a las interacciones que realizan los usuarios. La ontología considera información del registro de los eventos y actividades que los usuarios realizan en el campus virtual, como, por ejemplo, entregas realizadas, visualización de contenidos, intervenciones en foros, el rol de cada usuario en una asignatura, etc.

Se ha definido el mapeo entre los datos de origen y sus propiedades semánticas correspondientes en la ontología. También se han traducido los datos originales a formato RDF. De esta forma, se asegura la posible integración dentro de un único repositorio RDF de otras fuentes de datos de forma coherente, pudiendo consultarse fácilmente mediante consultas en lenguaje SPARQL.

Usando estos datos anotados semánticamente se han realizado una serie de análisis de la actividad que realizan los usuarios en el campus virtual. Este análisis se basa en series temporales del número de visualizaciones y del número de tareas entregadas. También se ha hecho uso de algoritmos de aprendizaje máquina para agrupar a los alumnos en diferentes tipos en base a la interacción que realizan con el campus virtual. Para finalizar la fase de análisis se ha entrenado un modelo predictor que permite predecir el grupo al que pertenece un alumno a través de los datos de su interacción con la plataforma.

La realización de este proyecto me ha permitido adquirir nuevos conocimientos en tecnologías relacionadas con la web semántica, como son: el diseño de ontologías, RDF, OWL, y el lenguaje de consultas SPARQL. He aprendido la sintaxis del lenguaje de programación Python y el uso de varias librerías para la representación de datos, análisis de datos usando machine learning, y conectores con bases de datos relacionales y semánticas. Además, me ha servido para recordar y volver a poner en práctica los conocimientos aprendidos en base de datos.

Para finalizar, creo que su realización ha contribuido a mejorar mi capacidad para enfrentarme a problemas reales y solucionar los problemas que se van presentando durante el desarrollo de un proyecto.

Como trabajo futuro, se propone ampliar el conjunto de datos con datos de otras titulaciones ofertadas en ofertadas por la Universidad de Málaga, para, una vez construido el conjunto de datos, usar distintos algoritmos de machine learning para realizar un análisis complejo de los datos, de forma que se puede generar una serie de funcionalidades para el análisis avanzado de los datos obtenidos a partir de la plataforma Moodle del Campus Virtual. Para ello, se propone desarrollar una serie de pilotos que, además de los análisis de datos, y como consecuencia de éstos, integren funcionalidades de alarma, recomendación y/o detección de eventos relevantes, como: rendimiento oscilante o muy variable de los alumnos, exámenes o tareas enviadas desde la misma IP y cuestionarios respondidos mucho más rápido que la media.

También se propone extender la ontología con conocimiento del dominio para estudiar cómo usar dicho conocimiento para mejorar los algoritmos de análisis.

6

Bibliografía

- Semantic Web*. (2015). Obtenido de <https://www.w3.org/standards/semanticweb/>
- Casali, A., & Gonzalez, H. (2018). *Razonadores*. Obtenido de https://www.dsi.fceia.unr.edu.ar/images/TIA-Protege_2018.pdf
- Documentation of scikit-learn*. (08 de 2019). Obtenido de <https://scikit-learn.org/stable/documentation.html>
- e-Learning*. (2007). Obtenido de <https://cfp.us.es/area-de-alumnos/e-learning>
- e-Learning. Definición y Características*. (2007). Obtenido de <https://cfp.us.es/e-learning-definicion-y-caracteristicas>
- García, L. (18 de 4 de 2015). *Infografía E-learning*. Obtenido de http://procomun.educalab.es/sites/default/files/e-learning_infografia.png
- Granda, J. L. (10 de 05 de 2008). *Anatomía de implementación de un LCMS basado en moodle*.
- Hayes, P. J., & Patel-Schneider, P. F. (25 de 02 de 2014). *RDF 1.1 Semantics*. Obtenido de <https://www.w3.org/TR/rdf11-mt/>
- matplotlib user's guide*. (26 de 08 de 2019). Obtenido de <https://matplotlib.org/users/index.html>
- McGuinness, D. L., & Harmelen, F. v. (10 de 2 de 2004). *OWL Web Ontology Language*. Obtenido de <http://www.w3.org/TR/owl-features/>
- MySQL 8.0 Reference Manual*. (08 de 2019). Obtenido de <https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html>
- Ontologies*. (2015). Obtenido de <https://www.w3.org/standards/semanticweb/ontology>

OWL 2 Web Ontology Language. (11 de 12 de 2012). Obtenido de <https://www.w3.org/TR/owl2-overview/>

Protégé wiki. (8 de 2019). Obtenido de <https://protegewiki.stanford.edu/wiki/>

SPARQL 1.1 Overview. (21 de 3 de 2013). Obtenido de <https://www.w3.org/TR/sparql11-overview/>

Stardog. (07 de 08 de 2019). Obtenido de <https://www.stardog.com/docs/>

The Pyramid Web Framework. (09 de 2019). Obtenido de <https://docs.pylonsproject.org/projects/pyramid/en/1.10-branch/>

The Python Language Reference. (8 de 2019). Obtenido de <https://docs.python.org/3/reference/>

7

Anexo

En este anexo se incorpora la descripción detallada de las entidades del modelo que forman el modelo relacional facilitado por el Servicio de Enseñanza Virtual de la UMA.

Columna	Tipo	Descripción
id	BIGINT(10)	Identificador asignado por el sistema.
assignment	BIGINT(10)	Identificador de la tarea a la que pertenece la entrega.
userid	BIGINT(10)	Identificador del usuario que realiza la entrega.
timecreated	BIGINT(10)	Marca temporal de la primera entrega.
timemodified	BIGINT(10)	Marca temporal de la última modificación que realizó el usuario.
status	VARCHAR(10)	Indica el estado en el que se encuentra la tarea, por ejemplo: borrador, enviado, etc.
groupid	BIGINT(10)	El identificador del grupo al que pertenece la entrega en caso de tratarse de una entrega grupal.
attemptnumber	BIGINT(10)	Se usa para registrar el número de intentos de entrega realizados.
latest	TINYINT(2)	Permite conocer rápidamente si se trata del último intento de entrega realizado, ya que en este caso el campo contiene el valor 1.

Tabla 14: Descripción de la entidad _assign_submission.

Columna	Tipo	Descripción
role	VARCHAR(255)	Indica el rol de un usuario, por ejemplo: profesor, estudiante, invitado, profesor sin permiso de edición.
userid	BIGINT(10)	Identificador del usuario.
courseid	BIGINT(10)	Identificador de la asignatura.

Tabla 15: Descripción de la entidad _users_enrolled.

Columna	Tipo	Descripción
id	BIGINT(10)	Identificador asignado por el sistema.
courseid	BIGINT(10)	Identificador de la asignatura a la que pertenece la tarea.
name	VARCHAR(255)	Nombre de la tarea.
intro	LONGTEXT(2147483647)	Descripción de la tarea.
introformat	SMALLINT(4)	Indica el formato del campo de descripción.
alwaysshowdescription	TINYINT(2)	Si es falso, la descripción de la tarea solo se muestra después de la fecha de apertura de la tarea, si es verdadero se muestra siempre.
nosubmissions	TINYINT(2)	Este campo es un caché para que permite a las páginas de Moodle distinguir los tipos de tareas.
submissiondrafts	TINYINT(2)	Si es verdadero, los envíos de tareas se considerarán borradores hasta que el alumno haga clic en el botón Enviar tarea.
sendnotifications	TINYINT(2)	Permite deshabilitar las notificaciones por correo electrónico relativas al módulo de tareas.
sendlatenotifications	TINYINT(2)	Permite habilitar las notificaciones para envíos tardíos.
duedate	BIGINT(10)	Fecha de entrega mostrada a los estudiantes.
allowsubmissionsfromdate	BIGINT(10)	Si está relleno, solo se admitirán entregas después de la fecha establecida.
grade	BIGINT(10)	Indica la máxima puntuación sobre la que se califica la tarea.
timemodified	BIGINT(10)	La marca temporal de la última modificación realizada en la configuración de la tarea.
requiresubmissionstatement	TINYINT(2)	Obliga al estudiante a confirmar para entregar la tarea.
completionsubmit	TINYINT(2)	Si este campo se establece a 1, la actividad se marcará automáticamente como completada una vez que el usuario envíe la tarea.
cutoffdate	BIGINT(10)	Fecha final de entrega de la tarea, después de esta fecha no se admitirán envíos.
teamssubmission	TINYINT(2)	Este campo indica si se trata de una tarea grupal.
requireallteammemberssubmit	TINYINT(2)	Si está habilitado, un envío no será aceptado hasta que todos los miembros del equipo lo hayan enviado.
teamssubmissiongroupingid	BIGINT(10)	Identificador de agrupación para obtener los equipos de la tarea.
blindmarking	TINYINT(2)	Oculto la calificación de la tarea, hasta que se habilite su publicación.
revealidentities	TINYINT(2)	Permite revelar las identidades de los estudiantes al calificar envíos de una tarea.
attemptreopenmethod	VARCHAR(10)	Determina cómo se permite a los estudiantes el reenvío de tareas.
maxattempts	MEDIUMINT(6)	Establece el número de intentos de entrega permitidos.
markingworkflow	TINYINT(2)	Permite activar el seguimiento del estado del proceso de calificación de la tarea.
markingallocation	TINYINT(2)	Permite habilitar la asignación de la evaluación de la tarea a un profesor en concreto.
sendstudentnotifications	TINYINT(2)	Permite enviar notificaciones al estudiante cuando se califica una tarea.

Tabla 16: Descripción de la entidad _assing.

Columna	Tipo	Descripción
id	BIGINT(19)	Identificador asignado por el sistema.
eventname	VARCHAR(255)	Campo calculado automáticamente copiando la ruta de la clase que genera el evento.
component	VARCHAR(100)	Campo calculado automáticamente con el nombre de la clase que genera el evento.
action	VARCHAR(100)	Tipo de acción que ha disparado el evento.
target	VARCHAR(100)	Etiqueta del tipo de objeto al que afecta el evento.
objecttable	VARCHAR(50)	Nombre de la tabla de la base de datos que representa al evento.
objectid	BIGINT(2)	Identificador del registro dentro de la tabla indicada en el campo objecttable.
crud	VARCHAR(1)	Este campo contiene una 'c' en caso de que se trate de insertar datos, una 'r' en caso de que se trate de una lectura, una 'u' en caso de que se trate de actualizar y una 'd' en caso de que se trate de un borrado.
edulevel	BIT	Bit que indica el grupo al que pertenece el usuario que ha disparado el evento sus posibles valores son: 1 para profesores, 2 para estudiantes y 0 para eventos que no tengan relevancia educativa.
contextid	BIGINT(19)	Identificador del contexto en el que se ha producido el evento.
contextlevel	BIGINT(19)	Indica si se produjo en una asignatura, en una tarea, etc.
contextinstanceid	BIGINT(19)	Según el nivel de contexto, puede ser la identificación de la asignatura, la identificación de un módulo, etc..
userid	BIGINT(19)	Indica el identificador del usuario que ha disparado el evento.
courseid	BIGINT(19)	Identificador de la asignatura en la que se ha disparado el evento.
relateduserid	BIGINT(19)	Si la acción implica a otro usuario, este campo contiene el id del otro usuario.
anonymous	BIT	Indica si se trata de un evento anónimo.
timecreated	BIGINT(19)	Marca temporal de cuándo se ha disparado el evento.
origin	VARCHAR(10)	Este campo indica el origen de la capa del cliente, si se trata de la interfaz web aparece web, si se trata de un servicio web aparece ws, si se lanza desde línea de comandos aparece cli.
realuserid	BIGINT(19)	Identificador real del usuario, cuando este tiene sesión iniciada en el sistema.

Tabla 17: Descripción de la entidad _logstore_standard_log.