Assignment 1 – Animal Scouter

University of Aveiro

Web Semântica

2022/2023

Group nº 2:

* João Bernardo Coelho Leite
* João Pedro dos Reis
* Luís Miguel Gomes Batista



Index

[Introduction 2](#_Toc130810009)

[Data, sources and transformations 3](#_Toc130810010)

[Dataset 3](#_Toc130810011)

[Selection of serialisation format 3](#_Toc130810012)

[Transformations 3](#_Toc130810013)

[Overview 3](#_Toc130810014)

[Data Operations (SPARQL) 4](#_Toc130810015)

[SELECT 4](#_Toc130810016)

[Get all animals from a given class 4](#_Toc130810017)

[Get all animals that produce a given nurturing 4](#_Toc130810018)

[Get all animals that produce both nurturing 4](#_Toc130810019)

[Get all animals that have a certain amount of legs 4](#_Toc130810020)

[Get all stored attributes of an animal 4](#_Toc130810021)

[ASK 4](#_Toc130810022)

[UPDATE 4](#_Toc130810023)

[Application Functionality 5](#_Toc130810024)

[Conclusions 6](#_Toc130810025)

[Application Configuration 7](#_Toc130810026)

[References 8](#_Toc130810027)

Introduction

The Semantic Web is an extension of the World Wide Web that allows data to be shared and reused across different applications, platforms, and systems. It is based on the principle of creating machine-readable data that can be easily understood and processed by computers. One of the key components of the Semantic Web is the use of linked data, which refers to a set of best practices for publishing and connecting structured data on the web.

In this report, we will discuss the conversion of a database from CSV (Comma Separated Variables) to N-Triples, a *Triplestore* database format, using a Python script. We will also explore the use of SPARQL, a query language used to retrieve and manipulate data stored in RDF formats. Additionally, we will showcase the features of a website that has been created to demonstrate the capabilities of the Semantic Web, including the ability to query and retrieve data using SPARQL.

The report will begin by describing the data and process of converting a database from CSV to N-Triples, and the advantages of using a *Triplestore* database. It will also be explained the basics of SPARQL and how it can be used to query and retrieve data from *Triplestore* databases.

Finally, the website that has been created to showcase the features of the Semantic Web will be discussed. The website will include a demonstration of how to use it, as well as examples of how the data can be visualized and analysed.

Overall, this report aims to provide an introduction to the Semantic Web, and to demonstrate the benefits of using linked data and *Triplestore* databases for managing and sharing structured data on the web.

Data, sources and transformations

## Dataset

The Zoo Animals dataset used in this Semantic Web analysis was sourced from Kaggle, a popular platform for data science and machine learning projects. The dataset consists of two separate CSV files, each containing 18 columns of data related to various attributes of different zoo animals.

### Selection of serialisation format

Before any transformations there was an important decision to make regarding what type of data should be used NT, N3 or RDF/XML. It ultimately was decided that NT (N-Triples) was the right choice for this project. The NT format is a widely accepted and standardized format for representing data in Semantic Web, it is designed to represent structured data providing ease of querying and analysis. Lastly, it’s very optimised for scalability being able to handle large amounts of data efficiently.

### Transformations

In order to prepare the dataset for use in the Semantic Web, several transformations were applied to the data. Firstly, the two CSV files were merged into a single file to simplify the data and make it easier to work with. Next, a script was created to automatically convert the merged dataset into the NT (N-Triples) format, which is a standard format for representing data in the Semantic Web. The script to generates unique IDs for the "class type", "animal name", and "type of nurturing" attributes in order to be able to name them separately to one another. These IDs served to create triples consisting of a subject, predicate, and object as per the example:

<http://zoo.org/animal/id/turtle> <http://zoo.org/pred/name> "Turtle" .

Example 1 Triple where a subject "id" has “name" which is "Turtle"

## Overview

After the data was converted to the NT format, it was loaded into a *Triplestore*, which is a type of database designed for storing RDF data. This allowed for efficient querying using SPARQL and analysis of the data, as well as integration with other Semantic Web tools and technologies such as GraphDB.

Overall, the use of the Zoo Animals dataset allowed for a robust and interesting analysis of the data using Semantic Web techniques. By applying transformations to the data and converting it into the NT format, it was possible to unlock valuable insights and information about the various attributes of different zoo animals.

Data Operations (SPARQL)

## SELECT

### Get all animals from a given class

Given a class ID as an object related to a subject animal ID, this same ID can be used to retrieve the name of the animal names related to this class type.

Text

Description automatically generated

Figure 1 Select query to get all animals from a category

### Get all animals that produce a given nurturing

Given a nurturing ID as an object related to a subject animal ID, this same ID can be used to retrieve the name of the animal names related to this nurture type of either “Eggs” or “Milk”.

Text

Description automatically generated with low confidence

Figure 2 Select query to get all animals that produce milk or eggs

If there are animals able to produce both types of nurturing a new query was made to intersect the previously mentioned queries and retrieve the name of all animals within that group.

Text

Description automatically generated

Figure 3 Select query to get all animals that produce eggs and milk

### Get all animals that have a certain amount of legs

Given a number of legs as an object related to a subject animal ID, this same ID can be used to retrieve the name of the animal names with this number of legs.

Timeline

Description automatically generated with low confidence

Figure 4 Select query to get all animals that have a specific number of legs

Since 0 values have not been stored, in order to select the animals that have no legs, all of them are selected and then a filter is used to find only the animals that don’t have that feature in relation to their respective ID. This, however, also retrieved class names and another filter had to be had to exclude those IDs.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 5 Select query to get all animals with no legs

### Get all stored attributes of an animal

Given the animals name its ID was used to retrieve all related data to that animal.

Timeline

Description automatically generated

Figure 6 Select query to get all attributes of a specific animal

For both nurture and class predicates, two extra queries have to be made to retrieve their names.

Text

Description automatically generated

Figure 7 Select query to retrieve name of class through ID

Text

Description automatically generated

Figure 8 Select query to retrieve name of nurture through ID

## ASK

### Get veracity of an attribute

Given an animal name and an attribute or feature it will return an Boolean result of whether that animal has a certain attribute or if it is a certain of a certain type.

Text

Description automatically generated

Figure 9 Ask query to verify or deny existence of an attribute

## UPDATE

### Insert

With an input of an animal name and a category selected, multiple other attributes can be selected as well to add a new entry to the dataset.

Text

Description automatically generated

Figure 10 Insert query to add a new entry to dataset

### Delete

This method will look for an animal ID in relation to a provided name, which, once identified, any and all predicates associated with this ID will be deleted.

Text

Description automatically generated with low confidence

Figure 11 Delete query to delete all entries related to an animal name

Application Functionality

Conclusions

Application Configuration

References