CAPITAL UNIVERSITY OF SCIENCE AND
TECHNOLOGY, ISLAMABAD

Handling Family Relations Inconsistencies in LoD (DBpedia)

by

Nouman Ahmad Khan

A thesis submitted in partial fulfillment for the
degree of Master of Science

in the

Faculty of Computing
Department of Computer Science

2020

I dedicate my dissertation work to my family, teachers and friends. A special feeling of gratitude is for my loving family for their love, endless support and encouragement.

CERTIFICATE OF APPROVAL

Handling Family Relations Inconsistencies in LoD (DBpedia)

by

Nouman Ahmad Khan

(MCS171031)

THESIS EXAMINING COMMITTEE

S. No.	Examiner	Name	Organization
(a)	External Examiner	Dr. Mansoor Ahmed	COMSATS University, ISB
(b)	Internal Examiner	Dr. Shahid Iqbal Malik	CUST, Islamabad
(c)	Supervisor	Dr. Muhammad Abdul Qadir	CUST, Islamabad

Dr. Muhammad Abdul Qadir
Thesis Supervisor
June, 2020

Dr. Nayyer Masood
Head
Dept. of Computer Science
June, 2020

Dr. Muhammad Abdul Qadir
Dean
Faculty of Computing
June, 2020

Author's Declaration

I, **Nouman Ahmad Khan** hereby state that my MS thesis titled "**Handling Family Relations Inconsistencies in LoD (DBpedia)**" is my own work and has not been submitted previously by me for taking any degree from Capital University of Science and Technology, Islamabad or anywhere else in the country/abroad.

At any time if my statement is found to be incorrect even after my graduation, the University has the right to withdraw my MS Degree.

(Nouman Ahmad Khan)
Registration No: MCS171031

Plagiarism Undertaking

I solemnly declare that research work presented in this thesis titled "**Handling Family Relations Inconsistencies in LoD (DBpedia)**" is solely my research work with no significant contribution from any other person. Small contribution/help wherever taken has been duly acknowledged and that complete thesis has been written by me.

I understand the zero tolerance policy of the HEC and Capital University of Science and Technology towards plagiarism. Therefore, I as an author of the above titled thesis declare that no portion of my thesis has been plagiarized and any material used as reference is properly referred/cited.

I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of MS Degree, the University reserves the right to withdraw/revoke my MS degree and that HEC and the University have the right to publish my name on the HEC/University website on which names of students are placed who submitted plagiarized work.

(Nouman Ahmad Khan)
Registration No: MCS171031

Acknowledgements

All worship and praise is for ALLAH (S.W.T), the creator of whole worlds. First and leading, I would like to say thanks to ALLAH (S.W.T) for providing me the strength, knowledge and blessings to complete this research work. Special thanks to respected supervisor Professor **Dr. Muhammad Abdul Qadir** for his assistance, valuable time, and guidance. I sincerely thank him for his support, encouragement and advice in the research area. He enabled me to develop an understanding of the subject. He has taught me, both consciously and unconsciously, how good experimental work is carried out. I would also like to thank all members of Semantics research group for their comments and feedback on my research work.

I am highly beholden to my family, and friends, for their assistance, support, and encouragement throughout the completion of this Master of Science degree. This all is due to love that they shower on me in every moment of my life.

I pray to ALLAH (S.W.T) to bestow me with true success in all fields, and shower knowledge upon me for the benefit of Mankind.

Ameen

(Nouman Ahmad Khan)

Registration No: MCS171031

Abstract

Semantic web data sources of Linked Open Data (LoD) like DBpedia are currently facing significant data inconsistencies. Research has shown that inconsistencies are still present in the LoD (DBpedia) because of two major reasons firstly, the data source like Wikipedia contains the human error in the data, resultantly LoD (DBpedia) shows inconsistent information. Second reason is, wrong conversion method (DBpedia Mappings), which extracts inconsistent information from Wikipedia to LoD (DBpedia). This study aims to determine how to overcome the inconsistent data problem in DBpedia family relation based information. Building on existing work on removal of data inconsistencies, it asks: What are the problems in the data extraction process? In addition, how can we remove the inconsistencies, and prepare a consistent data store?

Based on the literature review on DBpedia data Inconsistency and enhancement of the DBpedia Ontology, this study provided the improved conversion method of DBpedia Ontology, and removed the human error based data in the Wikipedia source. It also provides a solution, which helps to remove the certain data inconsistencies related to the family relation information in the LoD (DBpedia) data source. Several SPARQL queries are explicitly developed, to fetch the inconsistent data from the LoD (DBpedia), and results were carefully analyzed to extract the reason of the inconsistency. The study improved the existing DBpedia Ontology properties with help of analysis of errors fetched, and enriched DBpedia Ontology with family relations. Alongside, also provided a solution to update the erroneous data in the Wikipedia, to remove the inconsistent data in the Wikipedia. Analysis of the experiment, demonstrated that inconsistencies can be removed from DBpedia data source, by improving the family relation based DBpedia Ontology Mapping, and removing human errors from the Wikipedia data source. The result indicates that the human error based information in the Wikipedia data source, and wrong conversion methods have an impact on DBpedia data. On this basis, it is recommended to improve the DBpedia Ontology Mapping scheme, and remove the inconsistency from the Wikipedia data source.

Contents

List of Figures

List of Tables

Abbreviations

DBP	Properties not listed in DBpedia
DBO	Properties listed in DBpedia
FRO	Family Relation Ontology
ILP	Inductive Logical Programming
LoD	Linked Open Data
ML	Machine Learning
RDF	Resource Descriptive Framework
RDFS	Resource Descriptive Framework Schema
SPARQL	Protocol and RDF Query Language
URI	Universal Resource Identifier
UMBER	Upper Mapping and Binding Exchange Layer

Chapter 1

Introduction

1.1 Background

Data stored in form of Semantic information like Resource Data Framework (RDF) databases are publically accessible on the web server, as DBpedia [1], and Wiki-Data [2]. The data is extracted from the data sources like Wikipedia [1, 3], and placed into RDF Databases [4] fulfilling DBpedia Ontology [5] mapping rules.

Wikipedia is an open source encyclopedia, created and updated by volunteers [6]; it contains articles in multiple languages (English, Deutsch, and many other languages) [1], related to different entities (persons, company, electronics, etc.). Wikipedia contains the information mostly in the free text [7], but it also contains the structured information in the table known as infobox [8, 9], many articles contains an infobox table. An infobox table contains the most important information about the article like category of the article, picture or image related to article, latitude/ longitude for places, family details etc [1]. Every Wikipedia infobox template has a class assigned to it, in DBpedia. Every instance of the infobox is considered as the property, and is mapped on the respective property in the mappings of the DBpedia Ontology Classes [10].

DBpedia is a popular Linked Open Data (LoD) based structured database, which contains data in the form of Resource Descriptive Framework (RDF), extracted

from the Wikipedia data source [11]. The DBpedia database can be accessed live using the platform [12], to perform query and extract the data. These days, DBpedia is probably the greatest representative of LoD. First publically accessed DBpedia dataset was published in the 2007 and the latest was published in April 2016. The most recent release of the DBpedia dataset portrays 6 million entities [13] shown in table 1.1. DBpedia represents the information in RDF; the most recent release of DBpedia contains 9.5 Billion RDF triples, which are saved in multiple languages [13].

TABLE 1.1: DBpedia Entities Language

Language	Triples
English	1.3 Billion
Other Languages	5 Billion

DBpedia extracts the data from the Infobox table of the Wikipedia. DBpedia fetch the data with the help of the extraction framework [14] which contains Mappings (Class and Properties) available in the DBpedia Ontology. Every distinct template of Wikipedia Infobox has respective Mapping Class in the DBpedia Ontology [10]. In the Mappings of the DBpedia Ontology, every respective Wikipedia infobox property is allocated to its respective Mapping Property of the DBpedia Ontology [10], to fetch the correct data entered in the Wikipedia Infobox. To fetch all of the infobox data, we need to create the respective mapping property in the respective DBpedia Ontology for every Infobox information.

The DBpedia Ontology is a shallow, generic, and cross-domain, which is created manually, based on the most commonly used infoboxes within Wikipedia. DBpedia latest version which was released in 2016-04, has more than 5000 template mappings, (considering all languages) and even more number of infobox properties [10]. Every distinct language will have different respective mapping in the DBpedia Ontology. Primary purpose of creating a mapping is to map every Wikipedia infobox template to the DBpedia. Every key value property of the Wikipedia infobox is saved as the property in the mapping class of the DBpedia Ontology, every property of the DBpedia Ontology has constraints and details associated

to it, and Wikipedia data must fulfill them to be extracted, and displayed in the DBpedia data source.

DBpedia contains semantic data [15] regarding the family relations, existing between the instances, which has error in it. These errors occur due to two main reasons [11, 16], firstly the data which is extracted from the source is corrupt, while it was extracted, and secondly, while conversion of the data after fetching it from the data source (Wikipedia), it became inconsistent because of error in conversion method [17]. These issues lead to a critical concern that while answering the queries on these data sources, it often returns wrong facts. Due to these data inconsistencies, the semantic computing compromise its basic advantage of high precision, responding to the query.

In this study, we will extract the inconsistencies in the DBpedia conversion method by extracting the family relation data from DBpedia with help of SPARQL queries. DBpedia conversion method helps to extract the information from the Wikipedia Infobox, converts it to semantic data and save to the DBpedia data dump [18]. We will check the consistency of the data, extract the inconsistent data, and get the logical reason for the inconsistency and create list of inconsistencies which are present because of DBpedia, and Wikipeida, and update the conversion method by adding new rules or improving previous one, so that the data inconsistencies are removed. In addition to it, we will improve the data stored in the Wikipedia, by removing the human errors, due to which DBpedia conversion method fetches the erroneous information. An example of data inconsistency in the DBpedia is given below, in Figure 1.1.

FIGURE 1.1: DBpedia data inconsistency.

Our work will help to make LoD (DBpedia) more consistent related to family relations, hence querying family relations will produce consistent data. Also we will enrich the DBpedia Ontology with the family relation based Mappings which do not exist previously, and populate the DBpedia data dump with the family relation based data.

1.2 Problem Statement

DBpedia conversion method helps to extract information from Wikipedia, and store in DBpedia data dump in the form of semantic data. Existing conversion method contains inconsistencies related to the Family relation mappings resultantly family relation semantic data stored in publically available LoD (DBpedia) is not consistent, shown in Figure 1.1 as a result querying family relations does not produce correct results.

1.3 Research Question

Frequent amount of data stored in the RDF triple stores (DBpedia) is inconsistent due to the wrong data conversion method from source (Wikipedia) to the LoD (DBpedia), or the source (Wikipedia) data contains human error. This decreases the precision of the query on RDF based databases.

Research Question 1: How can we discover inconsistencies related with family relations, in RDF triple stores (LoD)?

Research Question 2: What are those inconsistencies?

Research Question 2: What are the consequences of these inconsistencies?

Research Question 4: What are the reasons of these inconsistencies?

Research Question 5: How can we remove the inconsistencies, and prepare a consistent data store?

Research Question 6: How to populate new defined Wikipedia infobox property data into DBpedia data dump?

1.4 Scope

There are numerous data inconsistencies available in LoD (DBpedia), which occur due to wrong conversion method (DBpedia mappings) of the data from the data source (Wikipedia) to the LoD (DBpedia). In this research, we will expose the data inconsistencies existing in the family relation based semantic data in LoD (DBpedia) by implementing the SPARQL based queries on the DBpedia data source. Afterwards, we will extract the inconsistent data and verify the data from the source (Wikipedia), either in case the facts are correct in the source; we will validate the conversion process of the data from data source (Wikipedia) to the LoD (DBpedia), and update the conversion method (DBpedia Mappings) to extract the correct facts from the data sources. In case the data is not correctly

added in the data source (Wikipedia), we will correct the structure of data in the data source, so that DBpedia conversion method extracts the correct information.

1.5 Purpose

Linked data sources like DBpedia are providing the platform [12] to perform queries, and extract facts from these databases, for searching operation. However, frequent family relation based facts extracted from these data sources are not correct; this compromises the basic advantage of the semantic web of provision of precise information. Purpose of this research work is to extract the family relation based data inconsistencies from LoD (DBpedia), and to remove those data inconsistencies with reasonable accuracy, so that querying Family Relations, produce precise results.

1.6 Thesis Organization

Rest of the thesis is organized as follows: second chapter presents the methodology. Third chapter describes the literature review, and the related work. Experiments for inconsistencies identification are described in fourth chapter. Fifth chapter presents the removing of inconsistencies from DBpedia Ontology, sixth chapter contains details related to Wikipedia Infobox inconsistencies removal. Seventh chapter represents evaluation of the experiments. Eight chapter is about conclusion and future directions of the study.

Chapter 2

Methodology

In this chapter, we are going to explain our methodology. Below listed steps demonstrate methodology to the problem identification and its solution.

2.1 Methodology Steps

LoD (DBpedia) inconsistencies occurs because of two main reason; error in DBpedia conversion method, and error in the Wikipedia Infobox, from where DBpedia fetch data. To handle those family relation based inconsistencies in DBpedia data source, we have performed few experiments, and removed the inconsistencies. The complete orderly methodology is describe below:

1. Literature Review for data inconsistencies in LoD (DBpedia).
2. Inconsistencies Identification in LoD (DBpedia) and Wikipedia.
    - 2.1 Write SPARQL to extract family relations data from LoD (DBpedia).
    - 2.2 Analysis of the results for inconsistencies due to DBpedia conversion process (DBpedia Mappings), or Wikipedia data inconsistency, by comparing the result of the DBpedia with the data source (Wikipedia).

2.3 Extract the data inconsistencies in the DBpedia, and Wikipedia Infobox.

3. Identify the classes, and properties of the DBpedia Ontology, associated with data inconsistencies.

   3.1 Analyze the DBpedia Ontology properties and classes.

   3.2 Modify the DBpedia Ontology properties and classes, so that it does not fetch the inconsistent information.

4. Discover the reason of the data inconsistency, in Wikipedia.

   4.1 Analyze the Wikipedia Infobox table data structure.

   4.2 Modify the Wikipedia Infobox data, so that error is removed from the data source.

5. To evaluate results, repeat the process from point 2 to point 5, to check if the data inconsistencies have been removed by comparing results extracted, before removing inconsistencies with the results extracted after removing inconsistencies.

6. Conclude the results and publish the findings.

Detailed methodology Diagram is shown in Figure 2.1, it reflects the imagery view of the bullets discussed above.

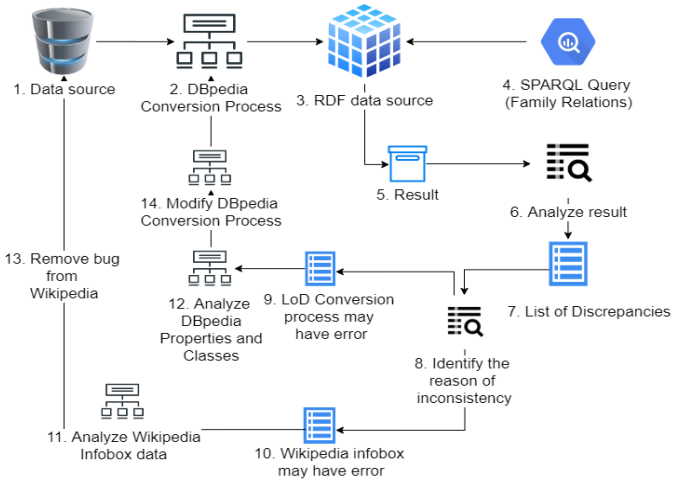

FIGURE 2.1: Methodology Diagram.

Chapter 3

Literature Review

3.1 Overview

The DBpedia project [1] is an effort of community to publish the data of Wikipedia infobox in the form of RDF. It has large volume of data and is relatively comprehensive. During the last few years, a great number of Linked Data datasets have been published in which entities are linked to their equivalent resource in DBpedia, making DBpedia a central interlinking hub for the Linked Data [19] datasets. The data consistency is an important metric for quality of ontologies. There might be inconsistencies in DBpedia. Error in data might exist in Wikipedia. From another viewpoint, when the data are extracted from Wikipedia, errors may occur during the conversion process [17]. For these inconsistencies, solutions have been provided, how ontology developers can keep away from, and users are capable to manage those inconsistencies.

Peron et al [20] did not worked on family relation based inconsistencies but they discovered inconsistencies in the DBpedia data source by comparing the domain and range of an object property. They defined domain inconsistency and range inconsistency, by comparing the occurrence of subject, and object resource of a property that does not belong to domain or range definition. These inconsistencies leads to proliferation of errors in DBpedia data, mainly related to data instance,

and their related ontology. The reason for existence of these inconsistencies is erroneous definition of domain, and range for the property. To overcome this issue, they introduced a method to quantitatively evaluate domain, and property misuse, and help fix those problems. For example, the resource dbpedia:The Beach Boys is not an instance of dbpedia:Person, but it uses the property dbpedia:hometown. By adding an inheritance relationship between dbpedia:The Beach Boys and dbpedia:Person may introduce errors on search related to dbpedia:Person.

In another work by Topper et al. [21], they worked on the inconsistencies focused by Hogan et al. [22], which are referring to accessibility, syntactical correctness, and consistency of published RDF data, they did not highlighted family relation inconsistency. They detected such semantic inconsistencies automatically by transforming semantic errors into logical ones by extending the axioms of the underlying ontology in order to cause a logical contradiction that can then be recognized by a reasoner. They considered syntactic errors in RDF that can be identified with the assistance of a RDF parser/validator, and syntactic inconsistencies in DBpedia, e.g. invalid date data. Because of existence of these inconsistencies, the data in the RDF database is not accurate, the domain and range of the properties are not properly fetched. For example, they extracted a fact that quotes Editoral Anagrama, that the novel 2666 has been published by Barcelona, which is obviously wrong, because novel cannot be published by Barcelona (city), as publisher should be Person or Company. The reason of existence of such inconsistencies are contradicting RDF triples. In order to remove these inconsistencies, they implemented new domain and range, and disjoint axioms.

Sheng et al. [17] discovered inconsistencies existing in the DBpedia information although they did not highlighted the family relation inconsistencies but they worked on general inconsistencies, which are defined as follow; usage of undefined class or properties. Secondly, usage of incompatible literals with ranges of data type properties, and third is, one class is subclass of, and disjoint with another class, and a class is subclass of two disjoint classes, lastly invalid entity definitions as members of disjoint classes. Due to these inconsistencies, DBpedia contains inconsistent data. For example, when they compared DBpedia information of

LibriVox with Wikipedia infobox, they figured out, it is an online digital library, while in DBpedia data source it is considered as a member of libraries in the real world. These inconsistencies exist because of use of wrong classes, and properties of DBpedia in the A Box of DBpedia. To remove those inconsistencies they conducted several experiments with UMBEL (Upper Mapping and Binding Exchange Layer) and DBpedia, by extending the disjoint axioms.

In another work by Cabrio et al. [23], have focused on the reasons for the production of inconsistent information. The maintenance of consistency plays an important role in DBpedia. It is important for sources posing queries to it. Family relation-based inconsistencies are not discussed by the authors. The inconsistencies include different information that is provided over a same query that might be giving identical, contradictory or differing answers while querying DBpedia multilingual chapters. The result set provided as a result of the query will not be consistent and different information will be provided by the same query. In the Web of Data, the combination of information related to a single object in the real-world is incorporated from different data sources, leading to inconsistency and raising questions over the quality of data itself. Research on how to prepare a consistent data source has been extensive and continuous. The authors have proposed an argumentation-based module that is integrated into QAKiS in order to provide unique and motivated answers to the user through careful reasoning and scoring.

Wienand and Paulheim [24] have presented the method of outliers detection to DBpedia. DBpedia is based on crowd-sourced contents and heuristic extraction methods, hence if a central hub of Linked Open Data and not free of errors. The paper does not discuss inconsistences related with family relations. The inconsistencies include both factual errors and problems during parsing such as number formats and issues with units of measurement that are no expected by the DBpedia extraction code. Since missing data and data that is wrongly assigned a type is present in the DBpedia, it makes DBpedia contain faulty types, hence wrong information and because data is entered manually from Wikipedia, it is prone to

errors. The reason of these inconsistencies is that DBpedia is based on heuristic extraction methods and crowd-sourced contents, hence it is not free of errors. Research on how to prepare a consistent data source has been extensive and continuous. The authors have proposed unsupervised outlier detection methods using Interquantile range (IQR), Kernal density functions (KDE) and various dispersion estimators that are combined with different semantic grouping methods such as outlier detection and clustering.

Similarly, Herradi et al. [25], presented new ontology, which efficiently represent relationships between person in a precise manner, and in different cultures/languages in order to be both, generic and adaptable to users. The author extracted the inconsistencies by comparing other existing ontologies. They analyzed that available ontologies does not have the relations defined in them, according to the cultures, due to which information is not accurately defined according to the different cultures, and the relationships are not properly defined. For example, in English, the term used to define a cousin relationship is cousin of. In French, there are two terms defining the relationship: cousin for male and cousine for female. In Arabic, it defines the cousin (male or female) from the side of the mother or the father. Therefore, there are eight terms defining the relationship of cousinship in Arabic. Reason for existence of such inconsistency is that ontologies are not enriched with culture level definitions. They analyzed different culture and generated concepts accordingly, and checked either the concept exist in each culture. If the concept exist in the respective culture then defined it in the ontology in precise manner using the definitions of the culture, which helped in populating the multi-language and multi-culture based meaningful data about the concept.

Rico et al. [26] proposed a technique to improve the results of the DBpedia SPARQL based query. In their work, they worked on the improving the results of the DBpedia, they did not worked on specifically family relations based queries. They stated, that users while querying in the DBpedia, do not use alternative properties in their queries. Consequently, users cannot get the authentic results of the query. They stated that DBpedia users do not usually use the dbp (properties not listed in DBpedia), in their queries, instead they use only dbo (properties listed

in DBpedia). To overcome the issue they proposed to use the dbp properties as alternative properties, so that the properties which are not available in the DBpedia. For example, to extract the birth place of a subject, users only considers dbo:birthPlace property, instead they must also include alternative properties i.e. dbp:birthPlcace, dbp:birhPlace, and dbp:birtPlace.

Piyawat et al. [27] proposed a method to correct the triple range violation error from DBpedia data dump. They did not highlighted family relation based inconsistencies. Range violation occurs when object of the triple, does not have the correct range type. For example, ¡dbr:Sedo, dbo:locationCountry, dbr:Cologne¿ this triple contains the range violation issue, because property (locationCountry) has the range Country while the object (Cologne) of above given triple is the City. Due to existence of range violation errors, DBpedia contains erroneous information, resulting in erroneous results in applications, which use DBpedia as a database. These inconsistencies occurs because of human errors in the Wikipedia Infobox table, and inconsistencies in the Infobox template. They corrected range violation in triples by finding the respective consistent object and replace it with the existing inconsistent object.

Rico et al. [10] proposed a method to remove the data inconsistencies by improving the mappings of the DBpedia Ontology. They discovered inconsistencies by comparing the RDF triples, which have same domain and range values by distinct property. They did not identified family relation based inconsistencies. In addition, they compared the triples, which have two distinct properties having the same value, e.g., largestCity and capital of a country, or birthPlace and deathPlace of a person could have same subject and same object pairs quite frequently even though those relations are not semantically equivalent. Because of these inconsistencies, a plethora of incorrect data is produced in the DBpedia data source. These inconsistencies exists because a diverse community of volunteers creates the DBpedia mappings, which are frequently wrong or inconsistent mappings. They proposed a data-driven method to detect the mapping deficiencies automatically with the help of machine learning (ML) techniques.

Martins et al. [28] presented a method to identify and reduce the inconsistencies in RDF dataset. The maintenance of consistency plays an important role in DBpedia. It is important for sources posting queries to it. They did not explicitly worked on Family relation-based inconsistencies. They identified triples that were not consistent with the Ontology in place. They used RDF triples to discover domain and range inconsistencies. These inconsistences can make DBpedia unreliable for use by any third party. Currently DBpedia is populated in two main ways: Firstly, through obtaining large dumps of data from Wikipedia that are altered periodically. Secondly, it is doing by using DBpedia live, which is a software to update DBpedia RDF triples in real time. However, the large dumps of data are infrequent because Wikipedia incorporates data from every type of source, reliable and unreliable, making it inconsistent. Research on how to prepare a consistent data source has been extensive and continuous. Various literatures have approached inconsistency classification, data co-evolution and various other methods. The authors have proposed an inconsistency detection method and an automatic correction method for DBpedia live updates.

Chen et al. [29] presented the method of automatic correction of erroneous assertions. Validity of assertions is an important consideration when it comes to investigating knowledge base quality. Family relation-based inconsistencies are not identified; however, inconsistencies can be identified by checking against soft property constraints mined from the global KB. Through checking consistency among logical constraints or rules, they detected the inconsistencies. A consistency score is obtained from the constraint-based validation checks that measures the degree of inconsistency among constraints. Wrong assertions such as ManchesterUnited and ManchesterCity, two football clubs that are based in Manchester, UK, can lead to facts about ManchesterUnited being incorrectly asserted about ManchesterCity and these are realized as inconsistencies. If the inconsistencies are not handled, the quality issues of KBs will increase and lead to less usefulness and usability of knowledge bases due to increase in erroneous assertions. Wikipedia is the main source of information of DBpedia and it consists of large number of contributors making contributions, leading to confusion and entity misuse. The

authors proposed a framework method for correcting entity misuse and correcting assertions whose objects are either erroneous entities or literals using ML techniques.

TABLE 3.1: Research Question Comparison

Paper	RQ1	RQ2	RQ3	RQ4	RQ5	RQ6
Peron et al. [20]	NO	YES	YES	YES	YES	NO
Topper et al. [21]	NO	YES	YES	YES	YES	NO
Sheng et al. [17]	NO	YES	YES	YES	YES	NO
Cabrio et al. [23]	NO	YES	YES	YES	YES	NO
Wienand and Paulheim. [24]	NO	YES	YES	YES	YES	NO
Herradi et al. [25]	YES	YES	YES	YES	YES	NO
Rico et al. [26]	NO	NO	NO	YES	YES	NO
Piyawat et al. [27]	NO	NO	NO	YES	YES	NO
Rico et al. [10]	NO	YES	YES	YES	YES	NO
Martins et al. [28]	NO	YES	YES	YES	YES	NO
Chen et al. [29]	NO	YES	YES	YES	YES	NO

Several type of inconsistencies are identified in the literature review. We performed survey shown in table 3.1 that compares the research work analyzed in the literature review, by using the metrics as our research questions.

We highlighted the strength, and weakness of the literature review shown in table 3.2 and 3.3.

TABLE 3.2: Strength and Weakness Comparison Table 1

Paper	Strength/ Algorithm	Weakness
Peron et al. [20]	Developed and evaluated a method designed to analyze a RDF graph and computes the domain/range inconsistencies.	Did not implemented the error diagnosis.
Tpper et al. [21]	Proposed an approach to enrich the Ontology by adding the axioms identified by Inductive Logical Programming.	The inconsistences were corrected manually using the statistical methods according to the suggestion.
Sheng et al. [17]	Defined 5 kinds of data inconsistencies in the form of rules.	Haven't focused on syntactic errors.
Cabrio et al. [23]	The study focuses on multilingual DBpedia instances.	A small amount of related work in the area of future work is discussed, lacking some direction about the further processing.
Wienand et al. [24]	The study focuses on DBpedia instances with numeric attributes. They also considered numeric attributes as numeric.	They are limited to integer and double type only. Not able to detect inconsistent data such as a wrong zip code.
Herradi et al. [25]	The study proposes an experimental framework that processes real world data. The paper is organized in a systematic manner.	The study had some user query limitations that lead to extraction of entities from a subset of DBpedia, and Freebase instead of using it entirely.
Rico et al. [26]	The study focuses on three DBpedia instances including English, Spanish and German.	A small amount of related work in the area is discussed. Testing of results with real users is not done, hence parameters selected can lead to some percentage of inaccuracy of results.

TABLE 3.3: Strength and Weakness Comparison Table 2

Paper	Strength/ Algorithm	Weakness
Piyawat et al. [27]	They corrected range violation errors of the existing data in the DBpedia data dump.	They did not evaluated the correctness of the triple.
Rico et al. [10]	The results are evaluated against carefully selected parameters.	Annotations require domain knowledge to decide if mapping is correct or incorrect so there is external dependency on domain knowledge expert.
Martins et al. [28]	Experimental model is well defined. Conclusion about cause-and-effect can be drawn.	Assuring validity of solution is important as the research is sample dependent, working on samples extracted from DBpedia. Different samples can lead to possible faulty interpretation of results.
Chen et al. [29]	This research worked on an important problem for KB curation which is assertion correction, and it is a problem that has rarely been studied. The experimental model is not dependent on KB metadata or external information and is an important contribution in the domain.	The impact of constraint-based validation and link prediction is variable and dependent on KB schema. Further impact varies from DBpedia to the Medical KB, thus limiting the scope of the effect of the techniques.

Previous work are not focused on evaluating the family relation based mappings. Literature survey infers that various methodologies are utilized to find inconsistencies in DBpedia. Most of them are focused on detecting the errors in data, or mappings that caused those errors. Herradi et al. [25] presented new Ontology to introduce the family relation concept, but instead of enriching the DBpedia

Ontology, they develop their own Ontology. Previous works did not enriched the DBpedia Ontology family relation mappings. Family relations were not analyzed by these approaches; therefore, we adopt an approach, which considers enriching the DBpedia Ontology with family relations mappings, and making the existing family relations in the DBpedia more consistent.

Chapter 4

Inconsistencies Identification in LoD (DBpedia) and Wikipedia

4.1 Overview

This chapter covers the detail of experiments, implemented on the data extracted from the LoD (DBpedia) and Wikipedia infobox to extract inconsistencies. Below listed items contains detail regarding inconsistencies extraction, and identification.

1. Develop the Family Relation based SPARQL queries.
2. Extracted the family relation based information from DBpedia data source.
3. Evaluated the results.
4. List down the data inconsistencies in the results.

We divided the work into 2 phases as shown in figure 4.1:

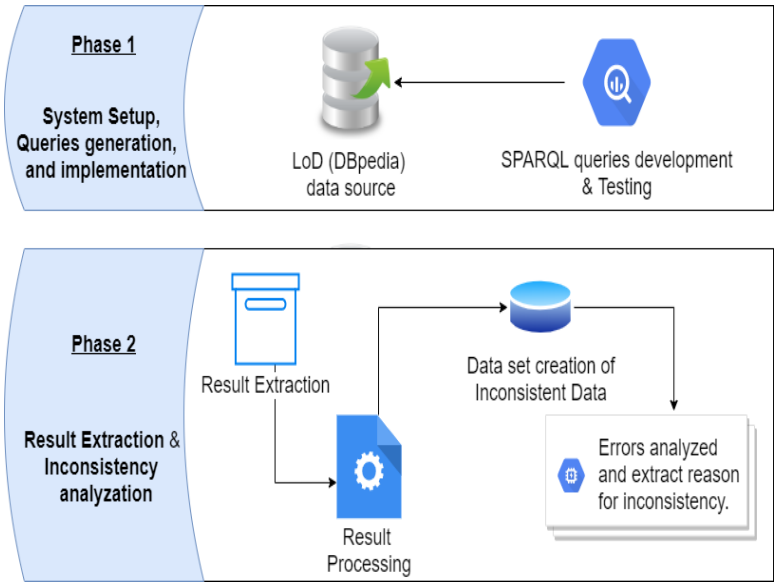

FIGURE 4.1: Detailed Experiment Diagram.

4.2 Phase 1: System Setup and Queries Generation

Development of SPARQL queries is the primary component of our system. The study relies on the results of the queries and by analyzing the results it fetch out, the modifications needed to be implemented in the DBpedia conversion method or the Wikipedia data structure. In this phase, we developed the SPARQL queries, and deployed them on the DBpedia SPARQL endpoint, which extracted the family relation, based data from the DBpedia data source, and analyzed the results.

DBpedia Extraction framework needs access to implement the modifications in the conversion method. We gained the access to the bureaucrat group (which is a group of researchers in the semantic computing, have access to improve the DBpedia Ontology conversion method). For development of the family relation queries, we need to check which relations exist in the DBpedia Ontology. We analyzed the DBpedia Ontology, and extracted the family relations, which exist in the Ontology. Then we developed the SPARQL queries based on those family relations, which do exist in the DBpedia Ontology. There were several relations, which had no data in the DBpedia, because their respective properties in the Wikipedia data source were not populated. In addition, there were several relations, which do not

exist in the DBpedia Ontology Mappings, but they do exist in the Family Relation Ontology (FRO).

TABLE 4.1: Relation that exist and have information in DBpedia data source

Relation that exist and have data in DBpedia source
Parent
Child
Spouse
Father
Mother

We proposed five queries, developed explicitly for extraction of inconsistent family relation based data. Following prefixes are used in development of all of queries.

PREFIX dbo: http://dbpedia.org/ontology/

PREFIX foaf: http://xmlns.com/foaf/0.1/

PREFIX rdfs: http://www.w3.org/2000/01/rdf-schema

PREFIX type: http://dbpedia.org/class/yago/

PREFIX dbp: http://dbpedia.org/property/

Query 1: Child death year is less than parent birth year

```
SELECT DISTINCT ?parent ?child ?ParentBirthYear ?ChildDeathYear
WHERE {
      ?parent a dbo:Person.
      ?child a dbo:Person.
          ?parent dbo:parent ?child.
          ?parent dbo:birthYear ?BD.
          ?child dbo:deathYear ?DD.
          BIND (year(xsd:date(?BD)) as ?ParentBirthYear).
          BIND (year(xsd:date(?DD)) as ?ChildDeathYear).
          FILTER (?ChildDeathYear< ?ParentBirthYear) }
```

FIGURE 4.2: SPARQL Query 1.

Query 2: Person A is child of Person B and vice versa

```
SELECT DISTINCT ?PersonA ?PersonB
WHERE {
     ?PersonA a dbo:Person .
     ?PersonB a dbo:Person .

     ?PersonA dbo:child ?PersonB.
     ?PersonB dbo:child ?PersonA.
}
```

FIGURE 4.3: SPARQL Query 2.

Query 3: Person A is parent of Person B and vice versa

```
SELECT DISTINCT ?person ?parents
WHERE {
       ?person a dbo:Person .
       ?parents a dbo:Person .

       ?person dbo:parent ?parents.
       ?parents dbo:parent ?person.

}
```

FIGURE 4.4: SPARQL Query 3.

Query 4: Person A is spouse of Person B, Person B is spouse of Person C, and Person C is spouse of A

```
SELECT DISTINCT ?personA ?personB ?personC
WHERE {
       ?personA a dbo:Person .
       ?personB a dbo:Person .
       ?personC a dbo:Person .

       ?personA dbo:spouse ?personB.
       ?personB dbo:spouse ?personC.
       ?personC dbo:spouse ?personA.

            FILTER (
            NOT EXISTS{
            ?personA owl:sameAs ?personC.
            })
}
```

FIGURE 4.5: SPARQL Query 4.

Query 5: Person A is spouse of itself

```
SELECT DISTINCT ?personA
WHERE {
      ?personA a dbo:Person.
      ?personA foaf:gender "male"@en.
      ?personA dbo:spouse ?personA.
}
```

FIGURE 4.6: SPARQL Query 5.

4.3 Phase 2: Result Extraction and Error Analyzed

In this phase, we extracted the results by implementing the SPARQL queries developed in phase 1. The results are analyzed in detailed. Firstly, the resultant data has been checked in the DBpedia source, to confirm results validity, and then the information is compared with the data source (Wikipedia). The property of DBpedia Ontology is compared with the respective Wikipedia infobox property, and then the value of infobox table property is checked. A data set is created for the information, which is inconsistent in the DBpedia and Wikipedia. After the creation of the dataset, the inconsistent data is analyzed carefully, and extracted the logical reason for the inconsistency which exist in both DBpedia, and data source (Wikipedia).

TABLE 4.2: Inconsistencies Analyzation

Query	Total Results	Examined Records	Error in data source (Wikipedia)	Error in LoD (DBpedia)
1	28	15	9	6
2	105	50	35	15
3	32	15	9	6
4	10001	30	23	7
5	5	5	5	0

In our results, we have extracted many data inconsistencies, existing because of wrong DBpedia conversion method. Consequently, the data in the DBpedia does not match with the data in Wikipedia infobox table. We have explained the data inconsistencies below in detail.

FIGURE 4.7: Richard Data in the DBpedia.

Figure 4.7 shows the data of the user Richard in the DBpedia, stating that Richard child is Barron Hilton, when we analyzed the data in the Wikipedia infobox, it was not correct.

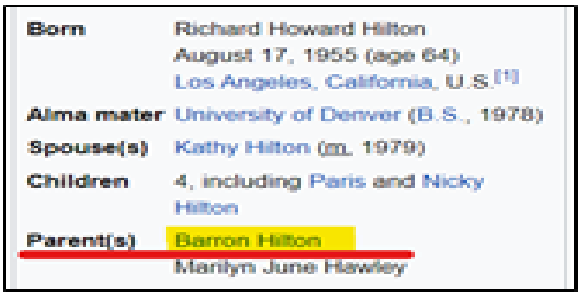

FIGURE 4.8: Richard Data in the Wiipedia.

As shown in Figure 4.8 in the Wikipedia, Barron was stated as parent of the Richard.

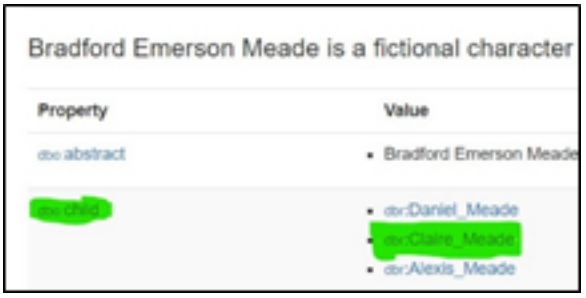

FIGURE 4.9: Bradford Data in the DBpedia.

In another example shown in Figure 4.9, we compared the data of the DBpedia with the Wikipedia Infobox. The data was also inconsistent in the DBpedia. In the DBpedia, Claire was displayed as the child of the Bradford. When we compared the result with the Wikipedia infobox, it was not correct.

FIGURE 4.10: Bradford Data in the Wikipedia.

In Wikipedia, Claire was added as the Spouse of the Bradford as shown in figure 4.10. The inconsistent data was fetched in the DBpedia because of inconsistent DBpedia conversion method.

FIGURE 4.11: Kirsten Data in the DBpedia.

One more example showed a data inconsistency, while analyzing the results shown in figure 4.11. We analyzed the family relation data in the DBpedia for the Kirsten, it showed Sandy, as child. When we compared the data with the Wikipedia Infobox, it was not correct.

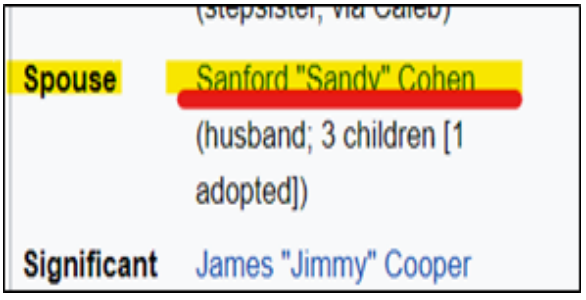

FIGURE 4.12: Kirsten Data in the Wikipedia.

In the Wikipedia, Sandy was declared as the spouse of the Kirsten as shown in figure 4.12. The inconsistency exist because of not existing of the detailed disjoint axioms in the DBpedia conversion process.

FIGURE 4.13: Ivar Data in the DBpedia.

In figure 4.13, we encountered another data inconsistency. When we analyzed the details of the Ivar family relations, it was stated in DBpedia that Arthur is child of Ivar. On comparing the results with the Wikipedia infobox, it was not correct.

FIGURE 4.14: Ivar Data in the Wikipedia.

As shown in figure 4.14, Arthur was parent of Ivar. DBpedia conversion process did not correctly fetch the data, resultantly DBpedia data source had inconsistent data, stating Arthur as child of the Ivar.

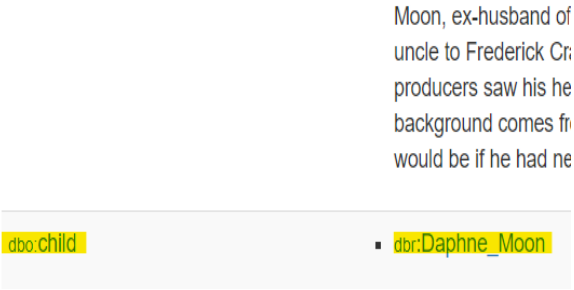

FIGURE 4.15: Niles Data in the DBpedia.

In another result, the data was not consistent. In DBpedia Daphne was shown as the child of Niles, as shown in figure 4.15. On comparing the results with the Infobox table, we encountered inconsistency in the data.

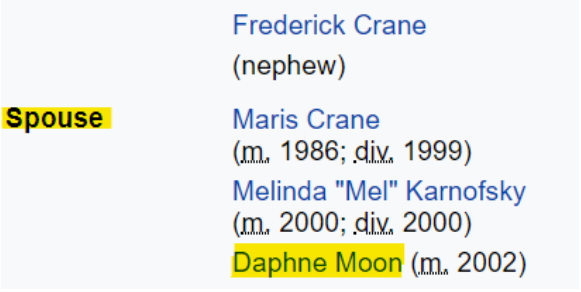

FIGURE 4.16: Niles Data in the Wikipedia.

In the Wikipedia Infobox Daphne was stated as the spouse of the Niles as shown inn figure 4.16. The data was inconsistent in the DBpedia, because of inconsistent DBpedia conversion process.

FIGURE 4.17: Jack Data in the DBpedia.

In another result, we analyzed the family relation details of Jack in the DBpedia, it contained the Angela as the child. On comparing the result with the Wikipedia infobox, we analyzed that it is inconsistent.

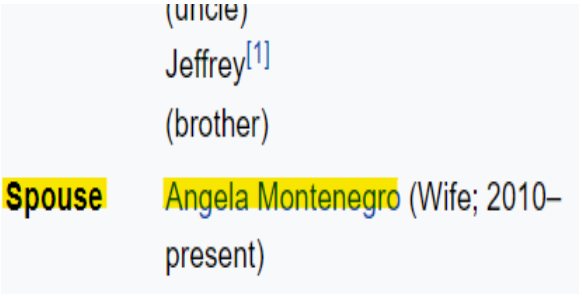

FIGURE 4.18: Jack Data in the Wikipedia.

In Wikipedia, Angela was added as the spouse of the Jack. The DBpedia conversion process was inconsistent, resulting in inconsistent data. We need to improve the DBpedia conversion process to get the consistent data.

FIGURE 4.19: Micheal Data in the DBpedia.

Figure 4.19 shows the result of the DBpedia query, which contained the inconsistent data. In the DBpedia Claudia was showed as the child of Jack. While on comparing the data with Wikipedia, it was incorrect.

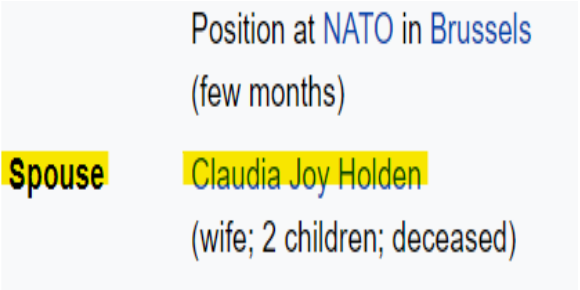

FIGURE 4.20: Micheal Data in the Wikipedia.

The Wikipedia infobox contains the information, stating Claudia as the spouse of the Michael. DBpedia contained the inconsistent information because of not including the proper disjoint axioms.

FIGURE 4.21: William Data in the DBpedia.

In another result extracted from the DBpedia, William was shown as the child of William, which could not be possible as the person cannot be the child of self, shown in figure 4.21. We compared the details in the Wikipedia infobox, and data was inconsistent.

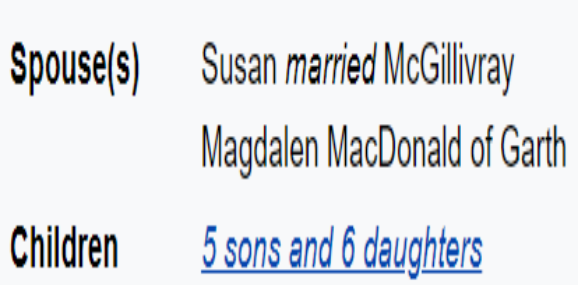

FIGURE 4.22: William Data in the Wikipedia.

The Wikipedia infobox table contained the quantify information of the children of William, instead of the William as child. Wikipedia infobox has inconsistent information stored in it.

FIGURE 4.23: Arwa bint Kurayz Data in the DBpedia.

In an example, shown in figure 4.23 we encountered data inconsistency in the Wikipedia data structure. The DBpedia contains the data Arwa bint Kurayz as child of Arwa bint Kurayz, as this could not be possible. On comparing the data with Wikipedia infobox, DBpedia contain inconsistent data.

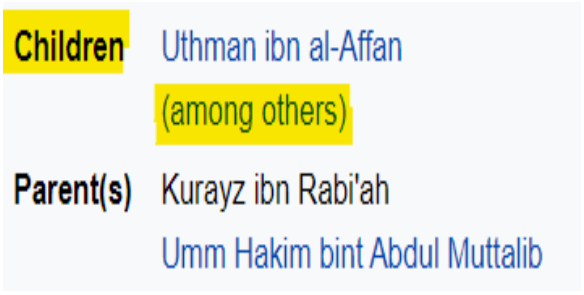

FIGURE 4.24: Arwa bint Kurayz Data in the Wikipedia.

As shown in figure 4.24, Wikipedia infobox contained the Uthman ibn al-Affan as the child, also it contained among others. Hence, data is inconsistent in the DBpedia data source.

FIGURE 4.25: Fatima bint Asad Data in the DBpedia.

In another result example shown in figure 4.25, we extracted the family relation details of Fatima bint Asad in DBpedia, it contained information, as Fatima is child of Fatima, which could not be possible. We analyzed the respective Wikipedia infobox.

FIGURE 4.26: Fatima bint Asad Data in the Wikipedia.

In the Wikipedia shown in figure 4.26, Fatima had child value as see below; the data was added inconsistent in the Wikipedia infobox, due to which inconsistency exist in the DBpedia.

FIGURE 4.27: Chloe Data in the DBpedia.

Figure 4.27 contains the DBpedia result representing the information of the Chloe, it shows the Buck, as the child of the Chloe, while after comparing results from the Wikipedia infobox, it is inconsistent.

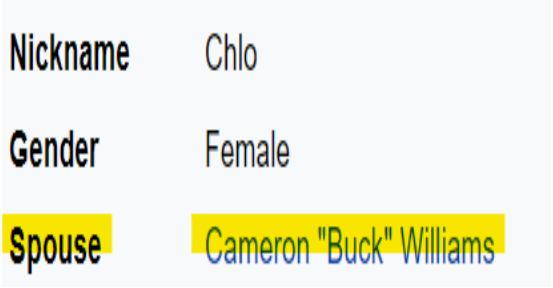

FIGURE 4.28: Chloe Data in the Wikipedia.

Wikipedia contain the Buck, as the spouse of the Chloe shown in figure 4.28, the data was inconsistent in the DBpedia because of not having required disjoint axiom in the property mapping.

After analyzing the inconsistencies, we have analyzed that the DBpedia mappings are not consistent; resultantly DBpedia contains the inconsistent data. To improve the DBpedia properties and classes, we need to analyze the domain and range of the properties. DBpedia mappings are open-source, user can create the mapping, because of not having the complete knowledge, not understanding the schema of info box, they often create inconsistent class or property mappings, and resultantly DBpedia contains inconsistent data.

Chapter 5

Removing Inconsistencies from DBpedia Ontology Classes and Properties

In this chapter, we will extract the inconsistencies in the DBpedia conversion process, and extract the reason for the existence of inconsistencies in family relation data. In addition to that, we will modify the inconsistent Property, or Class as shown in the Figure 5.1.

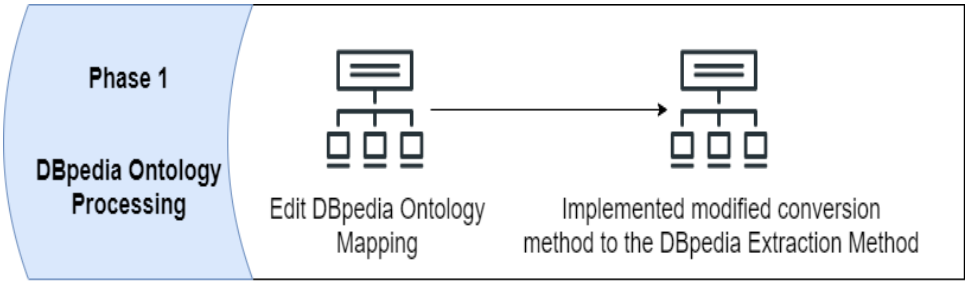

FIGURE 5.1: Process Flow Diagram.

5.1 DBpedia Ontology Inconsistencies

After careful analysis of the DBpedia data inconsistencies, we extracted the DBpedia Classes and Properties, which are associated with the inconsistent data. We analyzed the properties, and classes with the results extracted, and optimized them accordingly. We carefully validated the DBpedia Ontology Child property, and added the disjoint axiom with Spouse property, in addition with the Parent property. This helped DBpedia conversion method to not extract the father or mother name as child. While extracting data from Wikipedia to DBpedia, it compares the result of Child relation with the Spouse property and if it exists in it, then it does not fetch it. Resultantly, the data is consistent in the LoD (DBpedia).

```
{{ObjectProperty
| labels =
        {{label|en|child}}
        {{label|nl|kind}}
        {{label|de|Kind}}
        {{label|el|παιδί}}
        {{label|ja|子供}}
        {{label|ar|طفل}}
| rdfs:domain = Person
| rdfs:range = Person
| owl:equivalentProperty = schema:children
| owl:propertyDisjointWith = parent
| rdfs:subPropertyOf = dul:sameSettingAs
| owl:equivalentProperty = wikidata:P40
}}
```

FIGURE 5.2: Inconsistent Child Property.

In Figure 5.2, we have shown the inconsistent Child property of DBpedia Ontology. The Child property mapping does not contain the complete disjoint axioms, to fetch the consistent data. By adding the disjoint axiom with Spouse, we have removed the inconsistency in this property. In our experiments, we have highlighted the inconsistencies that exist because name of Father or Mother is written beneath Child as shown in Figure 5.2. The Wikipedia infobox contains the child property information, and it allows entering the father or mother of child alongside, which is not properly handled in the DBpedia Ontology Child property. We

have shown the Wikipedia infobox containing the Spouse name Sandy beneath Child name and relation with child in figure 5.2. Also the Spouse name is written in the Infobox Spouse Property.

FIGURE 5.3: Reason of Inconsistency in Child Property.

To overcome this issue, we critically analyzed the mapping of Child property, and added the disjoint with Spouse axiom. By adding this axiom, DBpedia Ontology will not fetch the name of the Father or Mother written beneath child name, as the disjoint with Spouse axiom will filter that inconsistency as shown in figure 5.4. We have added the disjoint with Spouse because father or mother of the child, would be spouse of the primary user (about whom the Wikipedia page is written).

Ontology object property (help)	
rdfs:label (en)	child
rdfs:label (nl)	kind
rdfs:label (de)	Kind
rdfs:label (el)	παιδί
rdfs:label (ja)	子供
rdfs:label (ar)	طفل
rdfs:domain	Person
rdfs:range	Person
rdf:type	
rdfs:subPropertyOf	dul:sameSettingAs
owl:equivalentProperty	wikidata:P40
owl:propertyDisjointWith	spouse

FIGURE 5.4: Optimized Child Property.

In the DBpedia, there were many inconsistent family relation based mappings, due to which data would not have been consistent; we also make those mappings more consistent.

To enrich the DBpedia family relations, we added the DBpedia ontology Classes: Man, and Woman. In absence of those classes, DBpedia data source contains the inconsistent data, as the DBpedia Ontology family relation properties were not properly defined, because of absence of above-mentioned Classes.

Ontology object property (help)	
rdfs:label (en)	mother
rdfs:label (de)	Mutter
rdfs:domain	Person
rdfs:range	Person
rdf:type	
rdfs:subPropertyOf	dul:sameSettingAs
owl:equivalentProperty	wikidata:P25
owl:propertyDisjointWith	

FIGURE 5.5: Inconsistent Mother property in DBpedia Ontology.

Such as, the property Mother has domain Person Class as shown in figure 5.5, which was incorrect as Person could be either man or woman, resultantly data fetched with it would not be completely consistent.

Ontology class (help)	
rdfs:label (en)	woman
rdfs:label (ru)	женщина
rdfs:label (pl)	kobieta
rdfs:label (fr)	femme
rdfs:label (da)	kvinde
rdfs:label (de)	Frauen
rdfs:label (ko)	여자
rdfs:label (ja)	女性
rdfs:label (nl)	vrouw
rdfs:label (it)	donna
rdfs:subClassOf	Person
owl:equivalentClass	wikidata:Q467
owl:disjointWith	

FIGURE 5.6: New added Woman Class in DBpedia Ontology.

As the Mother cannot be man, it must be Woman, so we defined new Class Woman as a sub Class of Person, and translated it into multiple languages to enrich the DBpedia in multiple languages as shown in figure 5.6.

Furthermore we updated the mother property domain as Woman. Previously the Woman class does not exist in the DBpedia Ontology, due to which we cannot specify the Mother domain as Woman, which results in inconsistent data. By adding the Woman class, DBpedia family relation inconsistency related to Mother will be removed as shown in figure 5.7.

Ontology object property (help)	
rdfs:label (en)	mother
rdfs:label (de)	Mutter
rdfs:domain	Woman
rdfs:range	Person
rdf:type	
rdfs:subPropertyOf	dul:sameSettingAs
owl:equivalentProperty	wikidata:P25
owl:propertyDisjointWith	

FIGURE 5.7: Mother property in DBpedia Ontology.

In addition to that, Father property mapping was also not consistent, as the domain of Father Property was Person, which could be either woman or man, resultantly the data fetched through father property could not be completely consistent as shown in figure 5.8.

Ontology object property (help)	
rdfs:label (en)	father
rdfs:label (de)	Vater
rdfs:domain	Person
rdfs:range	Person
rdf:type	
rdfs:subPropertyOf	dul:sameSettingAs
owl:equivalentProperty	wikidata:P22
owl:propertyDisjointWith	

FIGURE 5.8: Inconsistent Father Property.

Therefore, we defined Man Class as shown in figure 5.9 in the DBpedia Ontology. Man class is defined as a sub class of Person, and translated into multiple languages to facilitate DBpedia to fetch multi language data from the Wikipedia infobox.

Ontology class (help)	
rdfs:label (en)	man
rdfs:label (ru)	мужчина
rdfs:label (pl)	Mężczyzna
rdfs:label (fr)	Homme
rdfs:label (da)	Mand
rdfs:label (de)	Mann
rdfs:label (ko)	남자
rdfs:label (ja)	おとこ
rdfs:label (nl)	Mens
rdfs:label (it)	Uomo
rdfs:subClassOf	Person
owl:equivalentClass	wikidata:Q8441
owl:disjointWith	

FIGURE 5.9: New added Man Class in DBpedia Ontology.

In the mapping definition of the Father property, the domain of property was Person, which could result in inconsistent data, as Father must be Man.

To remove this inconsistency; we updated the Father Property domain with Man as shown in figure 5.10, so that data which would be fetched from the Wikipedia, would be consistent.

Ontology object property (help)	
rdfs:label (en)	father
rdfs:label (de)	Vater
rdfs:domain	Man
rdfs:range	Person
rdf:type	
rdfs:subPropertyOf	dul:sameSettingAs
owl:equivalentProperty	wikidata:P22
owl:propertyDisjointWith	

FIGURE 5.10: Improvement of inconsistent Father Property mapping.

5.2 Enriching DBpedia Ontology with Family Relations

DBpedia ontology only contain the Relative property, which does not explicitly define the family relation of the domain data with the range. For example, Jake Paltrow contains the relative based relation data in the DBpedia data source, but the relation is not explicitly defined in the DBpedia data source, as shown in figure 5.11.

FIGURE 5.11: Relatives of Jake Paltrow.

DBpedia ontology is not enrich with the family relation properties, due to which the DBpedia data source does not contains the family relations based enriched data. To enrich the DBpedia data source with family relations, we added new family relation based DBpedia Ontology Properties, which do not exist previously in the DBpedia ontology mappings, shown in table 5.1.

TABLE 5.1: Relations that do not exist previously in DBpedia Ontology

Relations that do not exist previously in DBpedia Ontology
Daughter
Son
Brother
Sister
Uncle
Aunt

We added the Brother relation Property in the DBpedia ontology, and translated the Brother Property in multiple languages, which are mostly in practice to populate the data in the Wikipedia infobox table, which are shown in table 5.2.

TABLE 5.2: Languages used in Wikipedia Infobox

Wikipedia Infobox input languages
English
Dutch
German
Greek
Japanese
Arabic

It is important to contain the translation of the property; to fetch the data consistently defined in multi culture based languages. We used our defined class Man as the domain, which help to make the Brother property more consistent. We added disjoint with parent so that, DBpedia conversion process do not fetch the inconsistent data which fetching data from the Wikipedia infobox table. Brother Property is shown in figure 5.12.

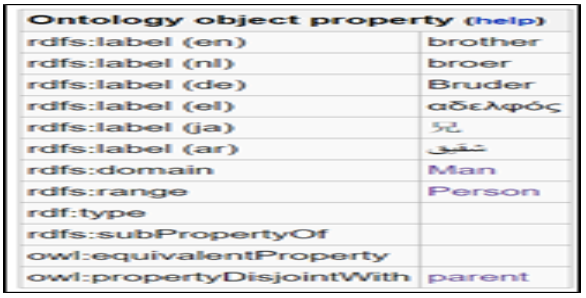

FIGURE 5.12: New added Brother Property in DBpedia Ontology.

In addition to that, we added Sister Property in the DBpedia Ontology to define the sister relation in the DBpedia data source. We translated the sister property in multiple languages shown in table 5.2, so that DBpedia can fetch the multi-language based data from Wikipedia infobox. We added the domain of the Sister relation as Woman, which is our newly defined DBpedia Ontology Class, to fetch

the consistent data from the Wikipedia infobox. Complete definition of Sister Property is shown in figure 5.13.

Ontology object property (help)	
rdfs:label (en)	sister
rdfs:label (nl)	zus
rdfs:label (de)	Schwester
rdfs:label (el)	αδελφή
rdfs:label (ja)	シスター
rdfs:label (ar)	اخت
rdfs:domain	Woman
rdfs:range	Person
rdf:type	
rdfs:subPropertyOf	
owl:equivalentProperty	
owl:propertyDisjointWith	parent

FIGURE 5.13: New added Sister Property in DBpedia Ontology.

In addition to that, we introduced the Son property in DBpedia ontology to enrich the family relation shown in figure 5.14. We translated the son property in multiple languages as shown in table 5.2. We added Man Class as a domain to Son property, to make the property mapping more consistent, and data fetched from the Wikipedia infobox is not erroneous, also we added disjoint with parent so that if data is inconsistent in the Wikipedia infobox table, it filter out in DBpedia data source.

Ontology object property (help)	
rdfs:label (en)	son
rdfs:label (nl)	zoon
rdfs:label (de)	Sohn
rdfs:label (el)	υιός
rdfs:label (ja)	息子
rdfs:label (ar)	ابن
rdfs:domain	Man
rdfs:range	Person
rdf:type	
rdfs:subPropertyOf	
owl:equivalentProperty	
owl:propertyDisjointWith	parent

FIGURE 5.14: New added Son Property in DBpedia Ontology.

Alongside we added the Daughter property in the DBpedia as shown in figure 5.15, translated in multiple languages defined in table 5.2. We added Woman (Class added by us) as a domain to Daughter Property, to make the data of DBpedia Ontology more consistent; also, we added the disjoint with parent, so that if Wikipedia data source contains the inconsistent data, it is filter out by the DBpedia conversion process.

Ontology object property (help)	
rdfs:label (en)	daughter
rdfs:label (nl)	dochter
rdfs:label (de)	Tochter
rdfs:label (el)	κόρη
rdfs:label (ja)	娘
rdfs:label (ar)	ابنة
rdfs:domain	Woman
rdfs:range	Person
rdf:type	
rdfs:subPropertyOf	
owl:equivalentProperty	
owl:propertyDisjointWith	parent

FIGURE 5.15: New added Daughter Property in DBpedia Ontology.

We also added Aunt Property shown in figure 5.16 in DBpedia Ontology, and translated it in multiple languages shown in table 5.2. Translating the property in multiple languages will help to make the conversion process more consistent while fetching data from multiple language based Wikipedia infobox tables. We used the Woman Class (defined by us) in the domain of the Property, so that consistent data is fetched, while the data conversion process from Wikipedia infobox to DBpedia. We added disjoint with Uncle property, to make the data more consistent. By adding this, DBpedia data source will contain data without inconsistency.

Ontology object property (help)	
rdfs:label (en)	aunt
rdfs:label (nl)	tante
rdfs:label (de)	Tante
rdfs:label (el)	θεία
rdfs:label (ja)	叔母
rdfs:label (ar)	عمة
rdfs:domain	Woman
rdfs:range	Person
rdf:type	
rdfs:subPropertyOf	
owl:equivalentProperty	
owl:propertyDisjointWith	Uncle

FIGURE 5.16: New added Aunt Property in DBpedia Ontology.

We enriched the DBpedia Ontology with family relation by adding with Uncle Property defined in figure 5.17, and translated it into multiple languages defined in table 5.2 to make the DBpedia data source more populated. As Wikipedia infoboxes are populated in multiple languages, and fetching data from the Wikipedia info box would require multi languages mapping. We added the domain as Man (created by us), to make the conversion process more consistent, and data in the DBpedia data source is error free, also we added disjoint with Aunt property, to filter out the inconsistent information while data conversion process from Wikipedia infobox to DBpedia data source.

Ontology object property (help)	
rdfs:label (en)	uncle
rdfs:label (nl)	oom
rdfs:label (de)	Onkel
rdfs:label (el)	θείος
rdfs:label (ja)	おじさん
rdfs:label (ar)	اخو الام
rdfs:domain	Man
rdfs:range	Person
rdf:type	
rdfs:subPropertyOf	
owl:equivalentProperty	
owl:propertyDisjointWith	Aunt

FIGURE 5.17: New added Uncle Property in DBpedia Ontology.

Conclusively, we enriched the DBpedia Ontology with the family relations. Previously DBpedia Ontology contain only relative property which was not enough to define the relation between person. Also we introduced two new Classes Man and Woman, to make family relations more precise and DBpedia Ontology more consistent. Previously there exists few relations in DBpedia which were not consistent because of missing Man and Woman Class.

5.3 Enriching DBpedia Data Source with Family Relation Data

DBpedia data dump does not contain the data related to the newly added family relations, because the data is fetched in to the DBpedia from two ways. First method to populate DBpedia data dump with family relations is to add family relation data into Wikipedia infobox. As Dbpedia extraction method extracts

data from Wikipedia infobox, hence it will fetch the newly added family relation data from respective Wikipedia infobox.

DBpedia extraction method extracts information as follow. DBpedia Class has respective Wikipedia infobox template, every Wikipedia infobox have specific properties, these properties are linked with DBpedia class properties. In this way Wikipedia is connected with DBpedia, and data is extracted from it.

On contrary to it, if WIikipedia infobox do not have property, but contain information in infobox as shown in figure 5.18

FIGURE 5.18: Wikipedia Relative Property.

In this scenario, we cannot explicitly extract specific data from the Wikipedia. As shown in figure 5.18, it contains relative property, and in its value it contains "Finneas O'Connell (brother)". In this case DBpedia extraction method will only extract the name of person i.e "Finneas O'Connell", and show in DBpedia as relative, instead of brother, because previously DBpedia didn't have the property Brother.

To handle this scenario we added several family relation properties in the DBpedia as shown in table 5.1. To explicitly populate the DBpedia data dump, we extracted the family relation data from the Wikipedia infobox, with the help of algorithm shown in figure 5.19.

```
import requests
import xlwt
from lxml import etree
from lxml import html
from xlwt import Workbook

# Workbook is created
wb = Workbook()
# add_sheet is used to create sheet.
sheet1 = wb.add_sheet('Sheet 1')

url='https://en.wikipedia.org/wiki/Billie_Eilish'
property="Relatives"

# fetching its url through requests module
req = requests.get(url)
store = etree.fromstring(req.text)

#getting primary person name
personnamequery = store.xpath('//*[@id="mw-content-text"]/div/table/tbody/tr[1]/th/div')
personname=personnamequery[0].text

#getting relation name
relationpersonnamequery = store.xpath('//*[@id="mw-content-text"]/div/table/tbody/tr[th/text()="'+property+'"]/td/a')
relationpersonname=relationpersonnamequery[0].text

#getting person relation with other person
personrelationquery = store.xpath('//*[@id="mw-content-text"]/div/table/tbody/tr[td/a/text()="'+relationpersonname+'"]/td/text()')
personrelation=personrelationquery[0]


personrelation=personrelation.replace("(", "")
personrelation=personrelation.replace(")", "")
personrelation=personrelation.replace(" ", "")
personrelation=personrelation.capitalize()

print("Name: "+personname)
print(property+": "+relationpersonname)
print("Relation: "+personrelation)

sheet1.write(0, 0, 'URL')
sheet1.write(0, 1, 'Person Name')
sheet1.write(0, 2, 'Relative')
sheet1.write(0, 3, 'Relation')

sheet1.write(1, 0, url)
sheet1.write(1, 1, personname)
sheet1.write(1, 2, relationpersonname)
sheet1.write(1, 3, personrelation)


wb.save('results.xls')
```

FIGURE 5.19: XPath XQuery Code to extract relation from Wikipedia.

For example, in figure 5.18, it contained "Brother" relation inside the "Relative" property, we extracted the relation, and relative name, and created a data set, which shows relation of person with relative explicitly as shown in table 5.3

TABLE 5.3: Family relation data set

URL	Person Name	Relative	Relation
www.dbpedia.org/resource/Sylvia Plath	Sylvia Plath	Finneas O'Connell	Brother
www.dbpedia.org/resource/Angelina Jolie	Angelina Jolie	Chip Taylor	Uncle
www.dbpedia.org/resource/Angelina Jolie	Angelina Jolie	Barry Voight	Uncle
www.dbpedia.org/resource/Angelina Jolie	Angelina Jolie	James Haven	Brother

After extracting the data set, we enriched the DBpedia data dump with the family relation data extracted, as shown in table 5.3. This helped DBpedia enriching the family relations.

Chapter 6

Wikipedia Infobox Inconsistencies Removal

In this chapter we will discuss the details regarding analyzing the Wikipedia Infobox data structure, and inconsistencies which exist in DBpedia because of inconsistent data in the Wikipedia Infobox. After fetching out the inconsistencies in the Wikipedia data structure, we will implement the necessary modification to remove the data inconsistency in the DBpedia. Process flow is shown in figure 6.1.

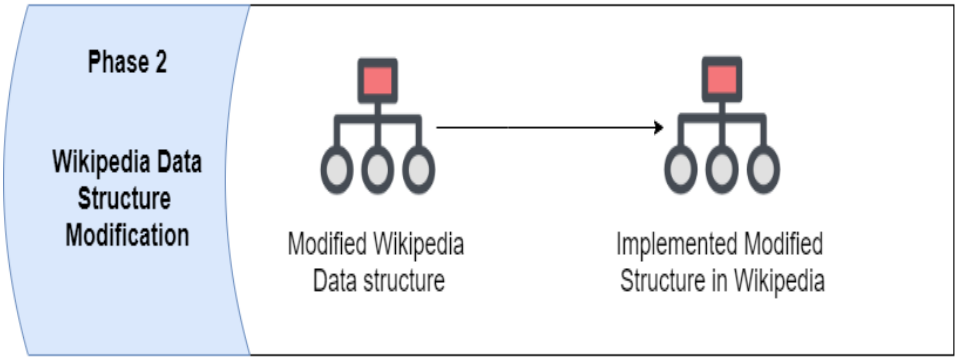

FIGURE 6.1: Process Flow Diagram.

6.1 Error Extraction

This chapter covers the detail of experiments, implemented on the LoD (DBpedia) data to extract the Wikipedia Infobox inconsistencies. Below listed items contains detail regarding development and implementation.

1. Implement the Family Relation based SPARQL queries (developed in Chapter 4), on DBpedia SPARQL endpoint, and extract the data.
2. Extracted the family relation based information from Wikipedia relative to DBpedia data source.
3. Evaluated the results.
4. List down the Wikipedia data inconsistencies shown in the information extracted in results.

In this chapter, we modified the Wikipedia infobox table structured data in the source, so that DBpedia mappings fetch the correct data, after fetching the logical reason for the existence of the data inconsistency. After modification of the Wikipedia data structure, we implemented the modified data in the Wikipedia. After the extraction of results from the DBpedia SPARQL endpoint, we analyzed the results, by comparing the DBpedia information with the Wikipedia infobox table details. After the careful analysis, we extracted several Wikipedia inconsistencies, which are resulting in erroneous information in the DBpedia data source.

In the figure 6.2 given below, user has entered the name of the primary person Fatima bint asad (about whom the infobox is written), in the children property of Wikipedia infobox. The children property was not showing the Fatima bint asad data in the front end, but it was saved in the backend, and displayed as hint on mouse hover. The DBpedia conversion method fetch the data from backend of Wikipedia infobox table and considered it as child, which resultantly made it inconsistent, as Fatima cannot be child of self.

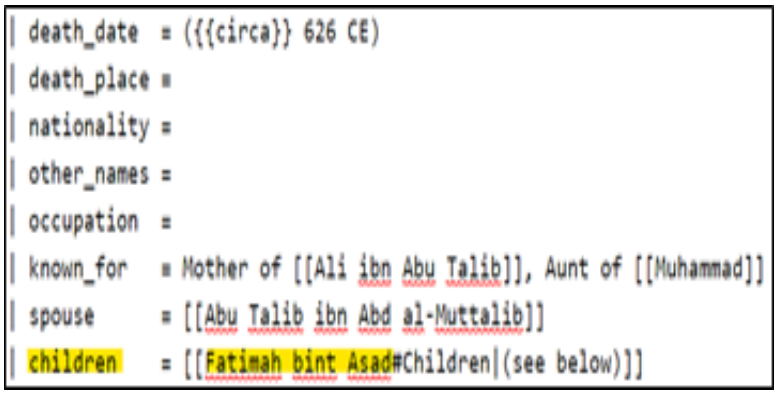

FIGURE 6.2: Wikipedia infobox inconsistency.

In another example shown in figure 6.3, the Wikipedia infobox contains the primary-user name in the spouse property value. But the primary user name was written in backend, such that it shows as a hint. DBpedia conversion method considered it also as a spouse.

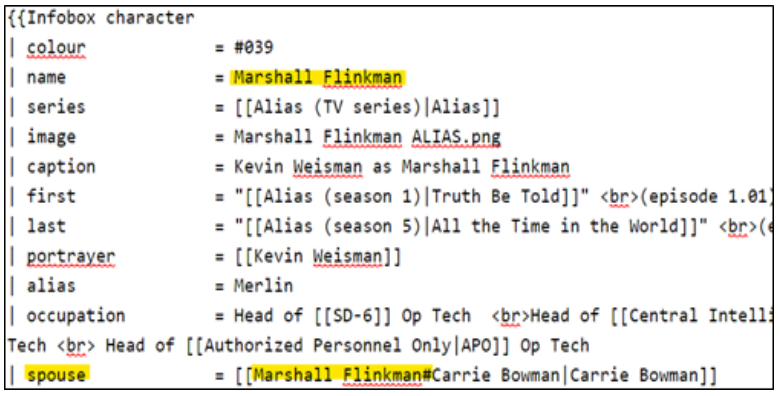

FIGURE 6.3: Wikipedia inconsistency example.

In another example of Wikipedia, the user inserted inconsistent data, resultantly, which created the inconsistency in the DBpedia data source as shown in figure 6.4.

FIGURE 6.4: DBpedia inconsistent data.

To remove this inconsistency we need to improve the data in the Wikipedia infobox. In the infobox the user wrote the name of the person (for whom the infobox was wrote), in the children property as a hint as shown in figure 6.5, which DBpedia conversion process fetched and considered as child. Resultantly, in the DBpedia the person Arwa bint Kurayz was considered as child of self (Arwa bint Kurayz).

```
{{Infobox person
| name        = Arwa bint Kurayz
| image       =
| alt         =
| caption     =
| birth_name  =
| birth_date  =
| birth_place =
| death_date  =
| death_place = [[Madina]]
| nationality =
| other_names =
| occupation  =
| known_for   = Mother of [[Uthman ibn Affan]], a [[Sahabah|companion]] of [[Muhammad]] a
| spouse      = [[Affan ibn Abi al-'As]]<br> [[Uqba ibn Abu Mu'ayt]]
| children    = [[Uthman ibn al-Affan]]<br>[[Arwa bint Kurayz#Children|(among others)]]
```

FIGURE 6.5: Inconsistent usage in Wikipedia.

We removed such inconsistencies, by fetching them with the help of SPARQL queries, and analyzing the result from the DBpedia, and compared the result with the Wikipedia infobox. We compared the data of both DBpedia and Wikipedia infobox, and logically fetch out the reason for the inconsistency, and removed it from the Wikipedia Infobox shown in figure 6.6.

```
{{Infobox person
| name        = Arwa bint Kurayz
| image       =
| alt         =
| caption     =
| birth_name  =
| birth_date  =
| birth_place =
| death_date  =
| death_place = [[Madina]]
| nationality =
| other_names =
| occupation  =
| known_for   = Mother of [[Uthman ibn Affan]], a [[Sahabah|companion]]
| spouse      = [[Affan ibn Abi al-'As]]<br> [[Uqba ibn Abu Mu'ayt]]
| children    = [[Uthman ibn al-Affan]]<br>[[#Children|(among others)]]
```

FIGURE 6.6: Consistent Wikipedia infobox.

This resultantly removed the erroneous data from the DBpedia data source. The result of the removal of inconsistency in Arwa bint Kurayz DBpedia information can be checked in the live DBpedia as shown in Figure 6.7.

FIGURE 6.7: Consistent DBpedia record.

In another example shown in figure 6.8, DBpedia contain the inconsistent record showing William as child of William (self), which could not be possible. We analyzed the results of DBpedia, and compared them with Wikipedia infobox.

FIGURE 6.8: Inconsistent DBpedia record.

We investigated the issue in the Wikipedia infobox, name of the primary user was written behind the code, which was visible as a hint not directly. DBpedia conversion process extract it and considered it as the child as shown in figure 6.9.

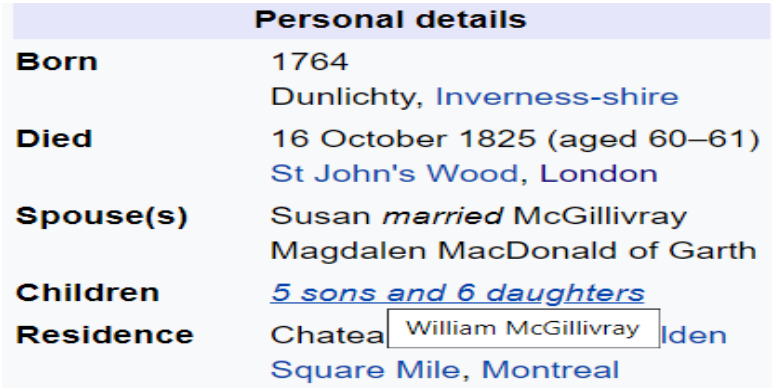

FIGURE 6.9: Inconsistent Wikipedia Infobox data.

The Wikipedia contained the record in quantity, and in the back code, it also contained the name of primary user, which was the cause of the data inconsistency in the DBpedia.

Table 6.1 contains results for every query applied on the SPARQL endpoint. It shows the number of errors, which exist in the LoD (DBpedia) data source because of erroneous Wikipedia data.

TABLE 6.1: Wikipedia Inconsistencies Comparison

Query No.	Total Results	Examined Records	Error in data source (Wikipedia)
1	28	15	9
2	105	20	12
3	32	15	9
4	10001	30	23
5	5	5	5

Conclusively, Wikipedia infobox while creation or update process must be filled carefully, and name of the Primary user should not be written in another different property. By handling this, DBpedia data related to family relations will be consistent. Or we can also create a user interface, using which person can update or create new Wikipedia page, as the chance of adding inconsistent details will be decreased.

Chapter 7

Evaluation of Experiments on Inconsistent LoD (DBpedia) Data

7.1 Overview

This chapter covers the detail of impact of experiments, implemented on the inconsistent information extracted from the DBpedia data source. After modifying the DBpedia Ontology Mappings, and improving the Wikipedia data structure, to remove the error in the DBpedia data as shown in Figure 7.1. Below listed items contains details regarding testing LoD (DBpedia) data.

1. Again, perform the test to extract the data from the data source (Wikipedia).
2. Compare the results with the previous results.

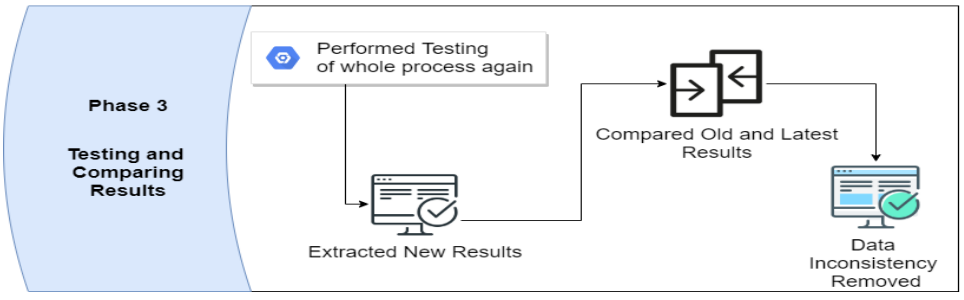

FIGURE 7.1: Flow Diagram.

7.2 Implementing Test Case

In this phase, we performed testing of whole process again, after implementing the modified DBpedia Ontology Mappings, and Wikipedia data structure. We implemented the same SPARQL family relation based queries, on the DBpedia SPARQL endpoint. We collected the results and compared them with previous fetched data set.

To test the results, we need to perform the queries and compare the latest results with the previous results. To perform the testing we design several test cases for queries, and run these test cases to analyze the results to prove if DBpedia family relation inconsistencies are removed.

7.2.1 Test Case 1:

In test case 1, we will validate the data consistency. To test the result of Query 2, open SPARQL query end-point, write the Query 2 as shown in figure 4.3. Then we need to compare the extracted results with the previous results. If resultant data is consistent that shows test case has passed. If not then test case is failed. Figure 7.2 shows Test case 1.

Test case ID		1	**Version 1.0**	
Test case Name		Consistent Query 2 Results		
Created by		Nouman Ahmad Khan		
Pre-Condition		Write SPARQL Query 2		
Post-Condition		Consistent results will be extracted.		
Step Detail		**Actual Result**	**Expected Result**	**Pass/Fail**
1	Open SPARQL Query end point	It will show consistent results, not containing the Spouse name in Child DBpedia Ontology property.	As per expectation	Pass
2	Execute Query 2			

FIGURE 7.2: Test Case 1.

7.2.2 Test Case 2:

In test case 2, we will validate the data consistency of Father Property. To test the result, open SPARQL query end-point, write the query related to Father Relation. Then we need to validate the extracted results. If resultant data is consistent that shows test case has passed. If not then test case is failed. Figure 7.3 shows Test case 2.

Test case ID		2	**Version 1.0**	
Test case Name		Father Property Consistent Records		
Created by		Nouman Ahmad Khan		
Pre-Condition		Write SPARQL Query related to Father.		
Post-Condition		Consistent results will be extracted.		
Step Detail		**Actual Result**	**Expected Result**	**Pass/Fail**
1	Open SPARQL Query end point	It will show consistent results, not containing Woman name in Father Property.	As per expectation	Pass
2	Execute Query containing Father Relation.			

FIGURE 7.3: Test Case 2.

7.2.3 Test Case 3:

In test case 3, we will validate the data consistency of Mother Property. To test the result, open SPARQL query end-point, write the query related to Mother Relation. Then we need to validate the extracted results. If resultant data is consistent that shows test case has passed. If not then test case is failed. Figure 7.4 shows Test case 3.

Test case ID		3	**Version 1.0**	
Test case Name		Mother Property Consistent Records		
Created by		Nouman Ahmad Khan		
Pre-Condition		Write SPARQL Query related to Mother.		
Post-Condition		Consistent results will be extracted.		
Step Detail		**Actual Result**	**Expected Result**	**Pass/Fail**
1	Open SPARQL Query end point	It will show consistent results, not containing the Man name in Mother Property.	As per expectation	Pass
2	Execute Query containing Mother Relation.			

FIGURE 7.4: Test Case 3.

7.2.4 Test Case 4:

In test case 4, we will validate the data consistency of DBpedia after removing inconsistency from Wikipedia data source. To test the result, open SPARQL query end-point, write the query 4 as shown in figure 4.5. Then we need to validate the extracted results. If resultant data is consistent that shows test case has passed. If not then test case is failed. Figure 7.5 shows Test case 4.

Test case ID		4	**Version 1.0**	
Test case Name		Wikipedia Inconsistency Removed		
Created by		Nouman Ahmad Khan		
Pre-Condition		Write SPARQL Query 4.		
Post-Condition		Consistent results will be extracted.		
Step Detail		**Actual Result**	**Expected Result**	**Pass/Fail**
1	Open SPARQL Query end point	It will show consistent results, i.e. not containing the records, which were inconsistent because of Wikipedia inconsistency.	As per expectation	Pass
2	Execute Query 4.			

FIGURE 7.5: Test Case 4.

After performing the test cases, we found very positive results. By removing the inconsistencies highlighted in Chapter 4, and 5, DBpedia data source became more consistent, which allows querying on DBpedia family relations more consistent. Also by enriching DBpedia Ontology with family relations, we can perform query on more detailed family relations. In addition to that, by adding the Man and Woman Class, DBpedia family relations are more consistent, to produce consistent results. Resultantly, DBpedia is enriched with the consistent family relations, and data related to them.

7.3 Results

After comparison, we had promising results, almost all of the data inconsistencies we highlighted are removed from the DBpedia data source. We elaborated an example of the results, the user added inconsistent data, resultantly, which created the inconsistency in the DBpedia data source as shown in figure 7.6.

FIGURE 7.6: DBpedia inconsistent record.

To remove this DBpedia inconsistency we modified the data in the Wikipedia infobox. This resultantly removed the erroneous data from the DBpedia data source. The result of the removal of inconsistency in Arwa bint Kurayz DBpedia information can be checked in the live DBpedia as shown in Figure 7.7.

FIGURE 7.7: Consistent DBpedia record.

In addition to that, we have also enriched the DBpedia with family relations. Previously, DBpedia contained the relative relationship, which does not explicitly shows the relation of the Person. Following are the family relations which exist previously in the DBpedia.

TABLE 7.1: Relation that exist in DBpedia

Relation that exist in DBpedia
Parent
Child
Spouse
Father
Mother

The family relation that exist in the DBpedia were inconsistent, we modified the existing family relations shown in Table 7.1, and made them consistent, resultantly we had improved result on querying DBpedia data source. In addition to that, we also enriched the DBpedia Ontology with new family relation Properties, and Classes shown in Table 7.2 and Table 7.3. This help to make DBpedia Ontology sparser and enrich regarding family relations.

TABLE 7.2: New defined family relation Properties in DBpedia Ontology

New defined family relation Properties in DBpedia Ontology
Daughter
Son
Brother
Sister
Uncle
Aunt

TABLE 7.3: New defined family relation Classes in DBpedia Ontology

New defined family relation Classes in DBpedia Ontology
Man
Woman

Before implementing experiments DBpedia Ontology contained 5 family relation based mappings as shown in table 7.1, alongside they were also inconsistent resulting in the inconsistent family relation dat in the DBpedia data dump. After implementing the experiments, currently we have 13 family relation which includes consistent existing mappings shown in table 7.1, and newly added mappings shown in table 7.2 and 7.3 which are also consistent. Family Relation Mappings comparison is shown in table 7.4

TABLE 7.4: Family Relation Mappings Comparison

Family Relation Mappings Existing Previously	Family Relation Mappings Existing Currently
5 (Inconsistent)	13 (Consistent)

Furthermore, we compared the previous results (extracted before implementing the modifications in the DBpedia Ontology Properties, and Classes), with the latest results (extracted after removal of inconsistency from DBpedia and Wikipedia), we had very good results, proving the removal of inconsistency from the DBpedia data source as shown in table 7.5.

TABLE 7.5: Results Comparison

Query No.	Inconsistent Records before removing inconsistencies	Inconsistent Records after removing inconsistencies
1	28	28
2	105	12
3	32	13
4	10001	10001
5	5	0

Results shown in 7.5, shows that after removing inconsistencies from the existing DBpedia Ontology Mappings, it produces consistent results.

In addition to that, we extracted the family relation based data from the authentic source (Wikipedia) with the help of python code shown in figure 5.19. We extracted family relation data from 168 Wikipedia Infobox Person entity, and sent to the DBpedia developers to populate the DBpedia data dump.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

We have analyzed from analytical and logical perspective, the DBpedia mappings that convert the unstructured family relation based data into linked data. The approaches given in this paper provides a solution to remove the family relation based inconsistencies from the linked data (DBpedia). In addition to that, we provide a method to improve the data structure for the Wikipedia infobox, to remove the human-error based inconsistency from LoD (DBpedia). Also we enriched the DBpedia Ontology by adding the family relations which do not exist previously in DBpedia Ontology.

Concerning these experiments, the primary problem is that DBpedia mappings does not contain the respective disjoint axioms, which would filter out the inconsistency while DBpedia data extraction process. For instance the attribute child or children is correctly mapped to the dbo:child, but frequent infobox data contains the information of parent under child stating as parent, which DBpedia extraction process consider as the child value also. However, in our approach it will apply the disjoint axiom on dbo:child property with the dbo:spouse, and it will filter out the inconsistent data. Though, if the mapping is corrected as suggested it will result in consistent data, as the data is filtered out by the disjoint axioms.

Another stimulating problem was that Wikipedia infobox data is not correct, because of human error while insertion of the infobox table details. For several instances, user add the name of the primary agent (about whom the article is), in the backend code which is not visible as direct information in infobox but resides in the backend, which DBpedia while conversion process extracts as the value for respective property in DBpedia Ontology. This problem can be overcome by providing the annotation along with infobox instance for not including the primary agent name or URI.

8.2 Future work

Our plan includes, to enrich the DBpedia Ontology with remaining family relation based Classes, and Properties defined in the Family Relation Ontology, so that the DBpedia conversion process can extract more family relation based semantic data from the data source (Wikipedia).

In addition to that, a user interface can be developed to enter the details in the Wikipedia Infobox; it will help to decrease the inconsistencies that arise because of inserting erroneous data in the Wikipedia infobox.

Also we can enrich DBpedia Ontology with the other existing Ontologies consistent mappings. In addition to that, we can remove the remaining inconsistencies highlighted in table 8.1.

TABLE 8.1: Inconsistencies removal future work

Query No.	Inconsistent Records after removing inconsistencies
1	28
2	12
3	13
4	10001

Bibliography

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: A nucleus for a web of open data." *ISWC/ASWC*, pp. 722–735, 2007.

[2] D. Vrandei and M. Krtzsch, "Wikidata: A free collaborative knowledgebase." *Commun. ACM*, p. 57(10), 2014.

[3] "Wikipedia [accessed 19-november-2019]." [Online]. Available: https://www.wikipedia.org/

[4] S. Hellmann, C. Stadler, J. Lehmann, and S. Auer, "Dbpedia live extraction." *LNCS*, pp. 1209–1223, 2009.

[5] "Dbpedia ontology [accessed 19-december-2019]." [Online]. Available: https://wiki.dbpedia.org/services-resources/ontology

[6] P. N. Mendes, M. Jakob, and C. Bizer, "Dbpedia: A multilingual cross-domain knowledge base." pp. 1813–1817, 2012.

[7] M. Volkel, M. Krotzsch, D. Vrandecic, H. Haller, and R. Studer, "Semantic wikipedia." *Proceedings of the 15th international conference on World Wide Web*, pp. 585–594, may 2006.

[8] D. Rinser, D. Lange, and F. Naumann, "Cross-lingual entity matching and infobox alignment in wikipedia." *Information Systems*, pp. 887–907, 2013.

[9] J. Lehmann, R. Isele, M. Jakobe, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer, "Dbpedia a

large-scale, multilingual knowledge base extracted from wikipedia." *Semantic Web*, pp. 167–195, 2015.

[10] M. Rico, N. Mihindukulasooriya, D. Kontokostas, H. Paulheim, S. Hellmann, and A. Gmez-Prez, "Predicting incorrect mappings: a data-driven approach applied to dbpedia." in *Proceedings of the 33rd annual ACM symposium on applied computing*. ACM, 2018, pp. 323–330.

[11] D. Caminhas, D. Cones, N. Hervieux, and D. Barbosa, "Detecting and correcting typing errors in dbpedia." *KDD 19 (DI2KG Workshop)*, Aug 2019.

[12] "Virtuoso sparql query editor [accessed 2-january-2020]." [Online]. Available: https://dbpedia.org/sparql

[13] "Dbpedia [accessed 23-november-2019]." [Online]. Available: https://en.wikipedia.org/wiki/DBpedia

[14] B. C. Graua, I. Horrocks, B. Motik, B. Parsia, P. Patel-Schneiderc, and U. Sattler, "Web semantics: Science, services and agents on the world wide web." *Journal of web semantics*, pp. 309–322, 2008.

[15] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web." pp. 34–43, May 2001.

[16] A. Dimou, D. Kontokostas, M. Freudenberg, R. Verborgh, J. Lehmann, E. Mannens, S. Hellmann, and R. V. de Walle, "Assessing and refining mappings to rdf to improve dataset quality," *ISWC*, pp. 133–149, 2015.

[17] Z. Sheng, X. Wang, H. Shi, and Z. Feng, "Checking and handling inconsistency of dbpedia," *International Conference on Web Information Systems and Mining*, pp. 480–488, 2012.

[18] M. Morsey, J. Lehmann, S. Auer, C. Stadler, and S. Hellmann, "Dbpedia and the live extraction of structured data from wikipedia." 2012.

[19] C. Bizer, T. Heath, and T. Berners-Lee, "Linked data: The story so far." *Semantic services, interoperability and web applications: emerging concepts.*, pp. 205–227, 2011.

[20] P. Youen, R. Frédéric, M. Gildas, and M. Pierre-François, "On the detection of inconsistencies in rdf data sets and their correction at ontological level," Jun. 2011. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00635854

[21] G. Topper, M. Knuth, and H. Sack, "Dbpedia ontology enrichment for inconsistency detection." in *Proceedings of the 8th International Conference on Semantic Systems*. ACM, 2012, pp. 33–40.

[22] A. Hogan, A. Harth, A. Passant, S. Decker, and A. Polleres, "Weaving the pedantic web." *LDOW2010*, p. 315, 2010.

[23] E. Cabrio, J. Cojan, S. Villata, and F. Gandon, "Hunting for inconsistencies in multilingual dbpedia with qakis." *International Semantic Web Conference*, pp. 69–72, 2013.

[24] D. Wienand and H. Paulheim, "Detecting incorrect numerical data in dbpedia." *European Semantic Web Conference*, pp. 504–518, 2014.

[25] N. Herradi, F. Hamdi, F. Ghorbel, E. Metais, N. Ellouze, and A. Soukane, "Dealing with family relationships in linked data." *International Workshop on Computational Intelligence for Multimedia Understanding (IWCIM)*, pp. 1–5, 2015.

[26] M. Rico, N. Mihindukulasooriya, and A. Gomez-Perez, "Data-driven rdf property semantic-equivalence detection using nlp techniques." *European Knowledge Acquisition Workshop*, pp. 797–804, 2016.

[27] P. Lertvittayakumjorn, N. Kertkeidkachorn, and R. Ichise, "Correcting range violation errors in dbpedia." *International Semantic Web Conference*, 2017.

[28] T. B. S. Martins and J. C. dos Reis, "Mechanism for inconsistency correction in the dbpedia live." *Technical report, Universidade Estadual de Campinas-UNICAMP*, 2019.

[29] J. Chen, X. Chen, I. Horrocks, E. J. Ruiz, and E. B. Myklebus, "Correcting knowledge base assertions." *arXiv preprint arXiv:2001.06917*, 2020.

Appendix A

Family Relation Ontology

Family Relation Ontology contains the family relations based property, and class mappings; graph is shown in figure below.

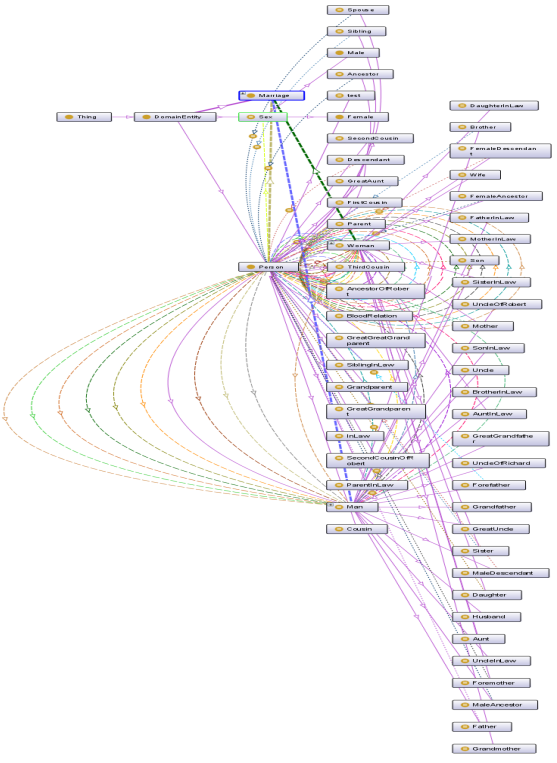

FIGURE A.1: Family Relation Ontology.

Appendix B

Method to Edit DBpedia Mappings

This section includes the information and methodology, regarding how to edit the DBpedia mappings. The complete process is defined below in steps:

1. Sign up in the DBpedia with email address.
2. After successful account creation, open the link given below: https://forum.dbpedia.org/t/mappings-wiki-account-requests/38
3. The link contains the DBpedia forum article; in the comment submit your username (created in point 1).
4. The DBpedia admin team will check your username, and add your username in the Bureaucrat group (Group of researchers working on DBpedia Ontology Improvement).
5. After that, you will be allowed to edit the DBpedia class, or property mappings.

Appendix C

Method to Edit Wikipedia Infobox

1. Sign up in the Wikipedia with email address.
2. After successful account creation, open the link given below: https://forum.dbpedia.org/t/mappings-wiki-account-requests/38
3. The link contains the DBpedia forum article; in the comment submit your username (created in point 1).
4. The DBpedia admin team will check your username, and add your username in the Bureaucrat group (Group of researchers working on DBpedia Ontology Improvement).
5. After that, you will be allowed to create new or edit the Wikipedia infobox template.

Appendix D

Code to fetch the data from Wikipedia

This section contains the python code, used to extract the explicit family relations from the Wikipedia infobox. The purpose of code is to extract the relative property data from the Wikipedia Infobox.

```
import requests
import xlwt
from lxml import etree
from lxml import html
from xlwt import Workbook

# Workbook is created
wb = Workbook()
# add_sheet is used to create sheet.
sheet1 = wb.add_sheet('Sheet 1')

url='https://en.wikipedia.org/wiki/Billie_Eilish'
property="Relatives"

# fetching its url through requests module
req = requests.get(url)
store = etree.fromstring(req.text)

#getting primary person name
personnamequery = store.xpath('//*[@id="mw-content-text"]/div/table/tbody/tr[1]/th/div')
personname=personnamequery[0].text

#getting relation name
relationpersonnamequery = store.xpath('//*[@id="mw-content-text"]/div/table/tbody/tr[th/text()="'+property+'"]/td/a')
relationpersonname=relationpersonnamequery[0].text

#getting person relation with other person
personrelationquery = store.xpath('//*[@id="mw-content-text"]/div/table/tbody/tr[td/a/text()="'+relationpersonname+'"]/td/text()')
personrelation=personrelationquery[0]


personrelation=personrelation.replace("(", "")
personrelation=personrelation.replace(")", "")
personrelation=personrelation.replace(" ", "")
personrelation=personrelation.capitalize()

print("Name: "+personname)
print(property+": "+relationpersonname)
print("Relation: "+personrelation)

sheet1.write(0, 0, 'URL')
sheet1.write(0, 1, 'Person Name')
sheet1.write(0, 2, 'Relative')
sheet1.write(0, 3, 'Relation')

sheet1.write(1, 0, url)
sheet1.write(1, 1, personname)
sheet1.write(1, 2, relationpersonname)
sheet1.write(1, 3, personrelation)



wb.save('results.xls')
```

FIGURE D.1: Code to fetch relation from the Wikipedia Infobox.