PUCRS

ESCOLA POLITÉCNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
DOUTORADO EM CIÊNCIA DA COMPUTAÇÃO

JÚLIA MARA COLLEONI COUTO

A MODEL FOR AUTOMATIZED DATA INTEGRATION IN HADOOP-BASED DATA LAKES

Porto Alegre
2022

PÓS-GRADUAÇÃO - *STRICTO SENSU*

Pontifícia Universidade Católica
do Rio Grande do Sul

PONTIFICAL CATHOLIC UNIVERSITY OF RIO GRANDE DO SUL
SCHOOL OF TECHNOLOGY
COMPUTER SCIENCE GRADUATE PROGRAM

A MODEL FOR AUTOMATIZED DATA INTEGRATION IN HADOOP-BASED DATA LAKES

JÚLIA MARA COLLEONI COUTO

Doctoral Thesis submitted to the Pontifical Catholic University of Rio Grande do Sul in partial fulfillment of the requirements for the degree of Ph. D. in Computer Science.

Advisor: Prof. Dr. Duncan Dubugras Alcoba Ruiz

Porto Alegre
2022

Ficha Catalográfica

C871m Couto, Júlia Mara Colleoni

A model for automatized data integration in Hadoop-based data lakes / Júlia Mara Colleoni Couto. – 2022.

108 p.

Tese (Doutorado) – Programa de Pós-Graduação em Ciência da Computação, PUCRS.

Orientador: Prof. Dr. Duncan Dubugras Alcoba Ruiz.

1. big data. 2. data lake. 3. Hadoop. 4. data profiling. 5. data integration. I. Ruiz, Duncan Dubugras Alcoba. II. Título.

Elaborada pelo Sistema de Geração Automática de Ficha Catalográfica da PUCRS
com os dados fornecidos pelo(a) autor(a).
Bibliotecária responsável: Clarissa Jesinska Selbach CRB-10/2051

JÚLIA MARA COLLEONI COUTO

A MODEL FOR AUTOMATIZED DATA INTEGRATION IN HADOOP-BASED DATA LAKES

This Doctoral Thesis has been submitted in partial fulfillment of the requirements for the degree of Ph. D. in Computer Science, of the Computer Science Graduate Program, School of Technology of the Pontifical Catholic University of Rio Grande do Sul

Sanctioned on March 31, 2022.

COMMITTEE MEMBERS:

Prof. Dr. Sérgio Lifschitz (DI/PUC-RIO)

Prof. Dr. Renata de Matos Galante (INF/UFRGS)

Prof. Dr. Tiago Coelho Ferreto (PPGCC/PUCRS)

Prof. Dr. Duncan Dubugras Alcoba Ruiz (PPGCC/PUCRS - Advisor)

I dedicate this work to my parents Getúlio (*in memorian*) and Eracilda Couto.

“If I have seen further, it is by standing upon the shoulders of giants.”
(Isaac Newton)

ACKNOWLEDGMENTS

First of all, I thank God and my spiritual guides for placing the right people on my road, for giving me the strength and the abilities that I need to fulfill my mission in this life. I am very grateful for my parents Getúlio Couto (in memoriam) and Eracilda Couto, my son Guilherme Couto, and my partner Ciro Oliveira for giving me so much love and support. I thank my sisters Dorália (in memoriam), Lisiane, Izaura, Ana, and the wonderful nephews and nieces that bring so much joy to my life. Thank you for understanding and accepting my absence whenever I could not participate in familiar events.

A very special thank to my advisor, Professor Duncan Ruiz, for being so generous in sharing your experience and knowledge with me, pointing me in the right direction, discussing ideas, and supporting my decisions. I could not have walked this path without your guidance. I also thank all the professors and secretary team of the PPGCC that have always helped me in the different stages of this thesis, specially Professor Sabrina Markzac, Professor Felipe Meneguzzi, and Regis Escobal.

To my beloved friends who were also my colleagues: Juliana Damásio, Olimar Borges, Tabajara Rodrigues, Carolina Toscani, Adriano Vogel, Vanessa Paixão-Cortes, I thank for the endless discussions on the most varied subjects that contributed so much for my academic and personal education. To my friends from elementary school: Lisiane Oliveira, Juliana Dias, Amélia Oliveira, Bruce Franklin, and so many others: i thank you so much for inspiring me and always being cheering for me.

I thank Professor Anil Wipat and his team at Newcastle University for the support and advice in the early stages of the thesis and for providing me with the hardware for first creating the data lake. Visiting the United Kingdom and researching in the NU was one of the most amazing experiences in my entire life.

I also thank to the committee members for the kindness and availability to read, review, and contribute to improving our work: Professors Tiago Coelho Ferreto, Sérgio Lifschitz, and Renata Galante. I really appreciate your valorous insights.

Finally, I want to acknowledge that this study was jointly financed by the *Coordenação de Aperfeiçoamento de Pessoal de Nivel Superior – Brasil (CAPES)* – Finance Code 001, SAP Labs Latin America, and Dell Brazil using incentives of the Brazilian Informatics Law (Law no 8.2.48, year 1991).

UM MODELO PARA INTEGRAÇÃO DE DADOS AUTOMATIZADA EM *DATA LAKES* BASEADO EM HADOOP

RESUMO

A imensa quantidade de dados que são gerados atualmente pelos nossos sistemas computacionais e dispositivos, conhecida por *big data*, requer tecnologias específicas, como *data lakes*, para que possam ser armazenados, processados e distribuídos. *Data lakes* são arquiteturas onde dados dos mais diversos formatos são armazenados para que sejam consultados quando necessário, sem a necessidade de esquemas prévios. *Data lakes* possibilitam o gerenciamento de ecossistemas de *big data*, e, hoje em dia, a maioria é criada tendo como base o *framework* Hadoop. Um dos desafios relacionados a *data lakes* é a integração dos dados de variados formatos. A integração dos dados é uma tarefa complexa que requer a atenção de um especialista, toma tempo e é sujeita a erros. Contudo, essa tarefa pode ser facilitada se forem utilizadas técnicas para conhecer o perfil dos dados. Nesta tese, desenvolve-se um modelo para automatizar o processo de integração de dados heterogêneos em *data lakes* baseados em Hadoop. O método desenhado para auxiliar a atingir os objetivos de pesquisa divide-se em 5 fases: Fundamentação, Implementação, Experimentação, Avaliação e Modelo final. As principais contribuições desta tese incluem os achados de três revisões sistemáticas da literatura, onde são exaustivamente discutidos os temas relacionados a *data lakes*, *big data profiling* e integração de dados em *data lakes*, e que serviram de base para o desenvolvimento de um modelo que possibilita a integração automatizada de dados heterogêneos em data lakes baseados no Hadoop, além dos experimentos com dados de bioinformática.

Palavras-Chave: *big data, data lake*, Hadoop, perfilagem de dados, integração de dados.

A MODEL FOR AUTOMATIZED DATA INTEGRATION IN HADOOP-BASED DATA LAKES

ABSTRACT

The massive amount of data currently generated by our computing systems and devices, known as big data, require specific technologies to be stored, processed, and distributed. Data lakes are architectures to store data of various formats to be queried when necessary, without needing a predefined schema. Data lakes aim to manage big data ecosystems, and most are currently created based on the Hadoop framework. A known challenge related to data lakes is integrating data from different formats. Data integration is a complex task that requires the attention of a specialist, besides being time-consuming and error-prone. However, this task can be facilitated if we use techniques to know the data profile. This thesis develops a model to automate the heterogeneous data integration process in Hadoop-based data lakes. In this sense, we design a method with five phases to help achieve the research objective: Foundation, Implementation, Experimentation, Evaluation, and Final Model. Our main contributions include the findings of three systematic literature reviews, where we deeply discuss themes related to data lakes, big data profiling, and data integration in data lakes, which served as a basis for the development of a model that enables the automatized integration of heterogeneous data in Hadoop-based data lakes, besides the experiments with bioinformatics data.

Keywords: big data, data lake, Hadoop, data profiling, data integration.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

API – Application Programming Interface

ASF – Apache Software Foundation

ATT – Attributes

AWS – Amazon Web Service

BDW – Big Data Warehousing

CCC – Confidence/Cost/Time

CERN – *Conseil Européen pour la Recherche Nucléaire* - European Organization for Nuclear Research)

CM4DL – Content Metadata for Data Lakes

CNPQ – *Conselho Nacional de Desenvolvimento Científico e Tecnológico* - National Council for Scientific and Technological Development

CPU – Central Process Unit

DAAL – Data as a Language

DDR3 – Double Data Rate 3

DF – Dataframes

DICE – Data Discovery by Example

ETL – Extract, Transform, and Load

FM-BIM – Facility Management-enabled Building Information Model

GB – Gigabyte

GDPR – General Data Protection Regulation

GFS – Google File System

GHZ – GigaHertz

HDDS – Hadoop Distributed Data Store

HDFS – Hadoop Distributed File System

HTTP – Hypertext Transfer Protocol

IDE – Integrated Development Environment

IID – Integrated Interactions Database

IOT – Internet of Things

JOSIE – JOining Search using Intersection Estimation

LGPD – *Lei Geral de Proteção de Dados* - General Law for Data Protection

LHC – Large Hadron Collider

MB – Megabyte

MS – Mapping Study

NAN – Not a Number

NCBI – National Center for Biotechnology Information

NOSQL – Not Only Structured Query Language

OBO – Open Biological and Biomedical Ontologies

OMIM – Online Mendelian Inheritance in Man

OSS – Open-Source Software

PICO – Population, Intervention, Comparison, and Outcome (PICO) or Population, Interest, and Context (PICo)

PMBOK – Project Management Body of Knowledge

POS – Point of Sale or Point of Service

PPGCC – *Programa de Pós-Graduação em Ciência da Computação* - Graduation Program in Computer Science

PRISMA-P – Preferred Reporting Items for Systematic Review and Meta-Analysis Protocols

RAM – Random Access Memory

RFID – Radio-Frequency Identification

RQ – Research Question

SLR – Systematic Literature Review

SNP – Single nucleotide polymorphisms

SO TGDS – Second-Order Tuple-Generating Dependencies

SQRE – Scalable Query Rewriting Engine

SSJOIN – Set Similarity Joins

START – State of the Art through Systematic Review

SWIM – Statistical Workload Injector for MapReduce

TB – Terabyte

TPC – Transaction Processing Performance Council

UML – Unified Modeling Language

USA – United States of America

XML – Extensible Markup Language

YARN – Yet Another Resource Negotiator

CONTENTS

1. INTRODUCTION

Our devices and computing systems are currently producing an unprecedented amount of data at a very high speed. Searching over the internet, we can easily find many examples of substantial data producers, such as Facebook, Netflix (Spangler, 2017), and the European Organization for Nuclear Research (CERN) (Le Meur and Tarocco, 2019). Indeed, the social network Facebook has over 2.89 billion users (Statista, 2021), and 1.91 million of them access their accounts daily (Noyes, 2021). According to the United Nations (UN, 2019), the current world population has surpassed 7.7 billion people; in that sense, almost one-third of the worlds' population may have a Facebook account.

Bioinformatics is another domain that generates enormous amounts of data - mainly after the next-generation genetic sequencing technologies' advent (Lesk, 2019). Since bioinformatics represents a particular and technical domain, the datasets are usually analyzed by domain experts. Still, as we can find a significant number of datasets available to be downloaded over the internet, it also arose the interest of beginners in data science. Those bioinformatic datasets can be used to analyze the most diverse scientific problems such as studying diseases, new drugs development, and drug repurposing.

This massive increase in data production encumbers the use of tools for data storing and processing since relational databases have difficulties processing such large volumes (Madden, 2012). In traditional storage models, we attempted to scale the servers by improving processing power and memory (vertical scalability), but it is so expensive that it is unfeasible for many companies. For this reason, people started to create cluster-based architectures with less powerful computers working in parallel (horizontal scalability). It is the origins of the distributed and high-scalable file systems, such as Google File System (GFS), its successor Colossus (Corbett et al., 2012), and Hadoop.

The Apache Hadoop framework has many advantages over other file systems. According to White (2015), besides being open-source, Hadoop also runs on commodity hardware, takes advantage of the parallel processing paradigm, and has high scalability. This way, Hadoop allows us to create a customized architecture since Hadoop is easily integrable with many other frameworks and tools. Actually, more than a hundred tools are available for the Hadoop Ecosystem (Roman, 2022), with thousands of possible combinations thereof.

Regarding big data architectures, data lakes are a trending topic. Data lakes can store raw data without previous modeling. Dixon (2010) created an analogy between data lakes and datamarts where he refers to a data lake as *water in its natural state, where people need to dive in to filter (or find information of interest)*, compared to a data mart, which would be a *packaged water bottle, clean and ready for consumption*. Nowadays,

the most widely used tool for creating data lakes is Apache Hadoop (Couto et al., 2019), a framework that eases big data management using off-the-shelf hardware.

One of the ways to present information about stored data is by generating data profiling. Data profiling creates data summaries of varied complexities, from simple counts, such as the number of records, to more complex statistics, such as functional data dependencies (Johnson, 2009). Data profiling allows us to better understand the data we have, and it is essential to help us choose the tools and techniques to process the data according to its characteristics. Data profiling is useful for query optimization, scientific data management, data analytics, cleaning, and data integration (Juddoo, 2015).

In this sense, data integration is a challenging task, even more nowadays, where we deal with the V's for big data, such as variety and volume (Searls, 2005; Lin et al., 2020). When we have different data types, the challenge is even more significant to perform interesting integrative queries over the distinct kinds of datasets. Furthermore, to analyze different datasets for data integration, a person must check the attributes and at least a dataset sample. To perform a more elaborated work, the person must look for the data dictionary of each dataset, and sometimes it is not easily available. According to Sawadogo and Darmont (2020), it is a problem since it is time-consuming, error-prone, and can lead to data inconsistency. Among the methods for data integration, the logic-based ones that consider the dataframes as sets, such as the based on the overlap of the values, could provide helpful solutions (Levy, 2000).

In this context, this thesis presents a model that performs automatized heterogeneous data integration, taking advantage of a data lake built based on Hadoop. According to ISO/IEC/IEEE (2010):

> *"A model is a representation of a real-world process, device, or concept; (...) A semantically closed abstraction of a system or a complete description of a system from a particular perspective."* (ISO/IEC/IEEE, 2010, p. 221)

Our model is based on a data lake that ingests, stores, processes, integrates, and helps visualize the data. Apache Nifi is responsible for data ingestion, and the HDFS (Hadoop Distributed File System) is responsible for data storage. Python is the base for data processing, and Neo4J creates the visualizations of the integrated data. The experiments are done with datasets of the bioinformatics domain. Therefore, this thesis covers the context of big data in architectures named data lakes that use the Hadoop framework, based on data profiling techniques to help the data integration task. Figure 1.1 illustrates a schema that visually correlates the research scope and keywords.

To perform this work, we developed a method based on 5 phases: Foundation, Implementation, Experimentation, Evaluation, and Final Model. In the Foundation Phase, we develop three literature reviews to better understand data lakes, big data profiling, and

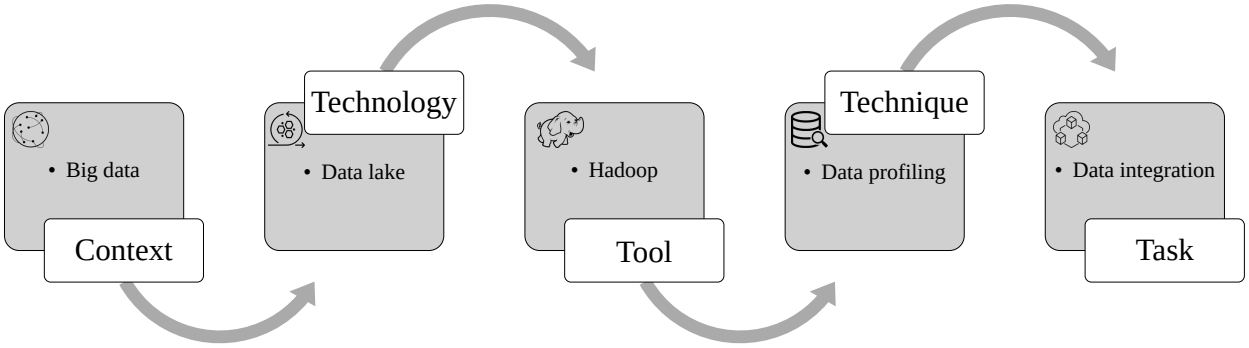


Figure 1.1: Correlation among research keywords
Source: The author (2022)

data integration in data lakes. Then, in the Implementation Phase, we configure the computational environment, choose the tools we will use, and create and implement a model for data integration in Hadoop-based data lakes. In the Experimentation Phase, we perform and report the experiments using bioinformatics datasets. Next Phase, we evaluate the model, and in the last Phase, we document the model alongside all the processes.

Our contributions include theoretical studies about data lakes, big data profiling, and data integration in data lakes, the model we developed based on the literature reviews results, and the experiments with bioinformatics datasets. Our model allows us to quickly ingest different kinds of textual datasets, transform them into dataframes, and, proposing an approach based on in-memory set similarity for data integration, we suggest the top-k points of integration for the data. We present experiments with eight bioinformatics datasets, and we compare our approach with manual data integration performed by a domain specialist.

1.1 Motivation

We have to use specific frameworks for big data manipulation, understanding, and integration due to the inherent characteristics of big data. Velocity, volume, and variety represent the most recognized aspects of big data. **Velocity** is related to the continuous amount of data produced at a high velocity and requires specific applications to handle this big data streaming. Apache frameworks such as Apache Flume, Apache Kafka, and Apache Nifi are among the systems designed for real-time data ingestion (Roman, 2022).

Volume represents the massive amount of data generated, and it requires the use of distributed file systems such as Hadoop HDFS (White, 2015). These systems also help reduce costs and increase scalability by enabling the creation of low-cost hardware computer clusters. A Hadoop cluster replicates data according to a scale, which is often equal to three. HDFS creates and updates metadata to keep the namespace up-to-date

every time data is ingested in a data lake. Metadata files record the location for each block, and when we have big files to store, the file can be split into many blocks, and the metadata system may have to process lots of entries.

The last characteristic, **variety**, is related to the various types of data stored in big data environments, such as structured, semi-structured, and unstructured data. Although traditional database management systems have evolved over the years to accept different data types, they work natively better with structured data. Consequently, NoSQL (Not Only SQL) databases have emerged to overcome the challenges related to storing semi-structured and unstructured data (Petricioli et al., 2021). According to Amghar et al. (2020), among the most used NoSQL databases, we can mention MongoDB, which stores data in document format; Neo4j, which works with graphs; and Apache Cassandra, which stores data in a columnar format. Although not being a database but a file system, HDFS itself is still the most used option for unstructured data.

Therefore, as we can associate several tools for creating hundreds of combinations of different systems to manipulate Terabytes of data, we must have efficient data integration. So, to extract any useful information from the data, it is imperative that we first know more analytics about this data, such as data distribution, where the most frequent data is, what are the outliers, and the number of unique values for each attribute.

1.2 Problem Statement

As stated by Khalid and Zimányi (2019), managing and querying a data lake is a difficult task, mainly because the data is heterogeneous, may have replicas or versions, have a considerable volume, and presents quality issues. In this sense, data integration, which represents 80-90% of the challenges for data scientists (Abadi et al., 2019), is a fundamental task to enable querying a data lake. However, integrating heterogeneous data into data lakes is a complex task, mainly due to the variety of data types (Hendler, 2014; Alrehamy and Walker, 2018) that can compose the data lake. If we only talk about textual data, there are countless extensions and possible formatting, such as: .txt, .docx, .csv, .xls, .xml, .json and so on.

Furthermore, the analysis for the integration depends on experts, often data scientists, who need to spend time inspecting data profile information, such as the types of each attribute, a sample of that data, or studying the data dictionary - when the dictionary is available. Finally, data integration is essential for extracting a more holistic view and information from the data lake, enabling us to make simple to complex queries and add value to the information.

1.2.1 Research Question

Based on the problem statement, below is our main research question:

> *How to create a model to automatize data integration for heterogeneous datasets taking advantage of the Hadoop-based data lake ecosystem?*

1.3 Objectives

This section presents the research's general and specific objectives based on the research question.

General Objective

We aim to develop a model to automatically integrate heterogeneous datasets in data lakes, taking advantage of the Apache Hadoop framework and its ecosystem.

Specific Objectives

We define the following specific goals:

1. Identify main concepts related to big data, data lakes, data profiling, data integration, and bioinformatics;
2. Identify the main system architectures to create data lakes;
3. Identify the state-of-the-art and challenges related to big data profiling;
4. Identify the state-of-the-art and challenges related to data integration in data lakes;
5. Create a model for automatized heterogeneous data integration of datasets in data lakes;
6. Evaluate the model;
7. Compare the model against state-of-the-art implementations;
8. Report the results to the scientific community.

1.4 Thesis Outline

In what follows, we investigate data integration in Hadoop-based data lakes. Chapter 2 presents the design of our research. Chapter 3 presents the theoretical background which is needed to understand our research. Chapter 4 presents the results of three systematic literature reviews we performed to recognize the related work and research opportunities related to our research. Chapter 5 introduces the model we developed for data integration in a Hadoop ecosystem, presents how we evaluate our model and discuss the results we achieved. Finally, Chapter 6 summarizes our contributions, limitations, publications, the availability of data and materials, and discuss future work.

2. RESEARCH DESIGN

This chapter details each step of the method we created to help us achieve our objective. We classified our research as experimental regarding technical procedures. In experimental research, the researchers insert new experimental variables into the environment and perform measures (Wazlawick, 2014). In our case, we add a data integration model in a Hadoop-based data lake and measure the results, comparing it to a specialist evaluation. We designed a method composed of five stages: Foundation, Implementation, Experimentation, Evaluation, and Final Model. We present the overall flow in Figure 2.1.

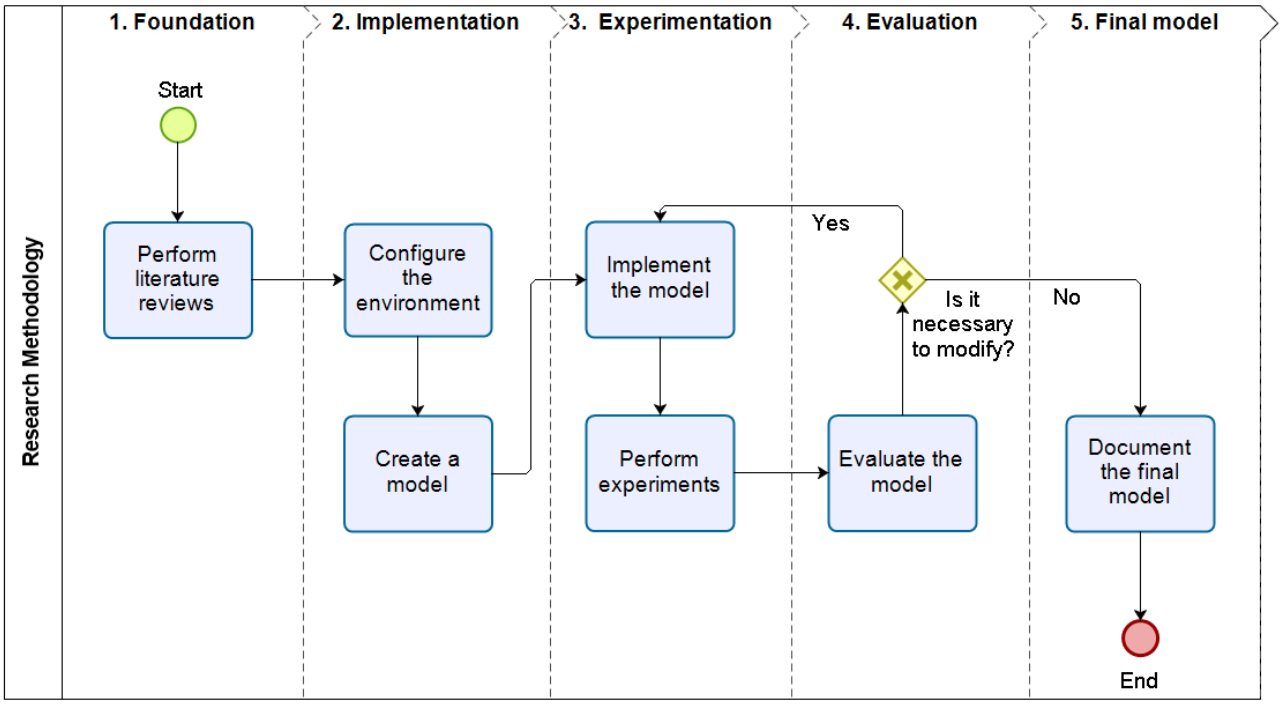


Figure 2.1: Research flow
Source: The author (2022)

2.1 Foundation

In the Foundation Phase, we *perform literature reviews*: one mapping study (MS) and two systematic literature reviews (SLR). An MS is a research method largely used to understand some subject's state of the art because it allows us to map its origins and how it developed over time, while an SLR enables us to deeply understand the context associated with a research subject (Brereton et al., 2007). We started by performing an MS to explore and understand how the data lake concept had evolved through time and also what are the techniques and tools that compose the most used architectures related

to data lakes. Then, we perform an SLR to delimit our search and understand how people perform big data profiling because profiling was little explored in the previous MS. Finally, we performed an SLR to deepen the knowledge about data integration, to map the most related studies.

2.2 Implementation

During the Implementation Phase, we configure the computational environment, choose the tools and techniques we use, and create a model for data integration in Hadoop-based data lakes. To *Configure the environment*, we select and prepare a software and hardware infrastructure to create a data lake, using bioinformatics datasets in a Linux-based environment. Next, we *create a model* responsible for data ingestion, storage, processing, and visualization.

2.3 Experimentation

In the Experimentation Phase, we design the algorithm for data integration, and we implement and test the model in the Python programming language. The *implement the model* task comprehends the actual development and tests of the system to manage data integration in Hadoop-based data lakes, besides data ingestion, storage, processing, and visualization. The next task is *to perform experiments* to test how our model can be compared with the baseline.

2.4 Evaluation

In the Evaluation Phase, we present how we asses our model. We *evaluate the model* against the manual mapping performed by a user specialist, and we present the runtime according to a scalability test. We also present the challenges we approach compared to the state-of-the-art presented in the literature.

2.5 Final Model

In the last Phase, we *document the final model* alongside all the processes that allows us to create the model, by writing scientific papers, submitting them to conferences and journals, reviewing the studies, and writing the thesis itself.

Remarks

The current chapter presented the research flow we designed and followed during the development of our work. We start by explaining the foundation and how we performed literature reviews, then we explained the implementation process, with the environment configuration and creation of the model. Next, we defined the experimentation process with the model development and experimentation. Finally, we presented the definition for the evaluation process and how we documented the final model.

3. BACKGROUND

In this chapter, we present the theoretical background related to this thesis. We address the following topics: big data, data lake, data profiling, data integration, and bioinformatics.

3.1 Big Data

Big data is a massive data set with large, varied, and complex structures, where we have difficulties for storage, analysis, and visualization for further processing (Sagiroglu and Sinanc, 2013). Big data is also essential for data science since it provides a vast amount of data to be analyzed. Big data has some specific characteristics, and the most known are volume, velocity, and variety. Farther, Taleb et al. (2019) compiled the following big data characteristics:

- Volume: the data size to be processed and analyzed.
- Velocity: the rate of use and transfer of data.
- Variety: different formats and types of data.
- Veracity: the accuracy of the data analysis results.
- Value: the aggregate value and contribution results from analyzing and processing the data.
- Variability: how much the data changes as time passes.
- Viscosity: how difficult it is to work with data.
- Volatility: related to data usefulness - eventually, some data can be not useful anymore.
- Validity: the data must be reliable and verifiable.
- Visualization: tools and techniques to visually understand the data.
- Virality: the spread rate and data trends.
- Vulnerability: concerning data security.

The contexts that use big data lately are the most varied, and the scientific literature presents some classifications for these contexts. For example, Sagiroglu and Sinanc (2013) lists the following: astronomy, atmospheric science, genomics, biogeochemistry,

biological science, life sciences, medical records, scientific research, government, natural disaster, resource management, private sector, surveillance military, financial services, retail, social networking, weblogs, text, documents, photographs, audio, video, search indexing, call detail records, Point of Sale or Point of Service (POS) information, Radio-Frequency Identification (RFID), mobile phones, sensor networks, and telecommunications. To complement, Brown et al. (2011) specified five main domains related to big data:

1. Healthcare: hospital management systems, individual analyses applied to patient profiles, personalized medicine, disease pattern analysis, improvements in public health.
2. Public sector: creating management transparency through data availability, needs discovery, performance improvement, decision-making assistance.
3. Retail: variety and price optimization based on consumer behavior analysis, logistics and distribution optimization, optimization of the physical location of products within stores.
4. Manufacturing: sales support, internet search-based applications, supply chain planning, improved demand forecasting.
5. Personal Location Data: intelligent routing, geo-localized advertising, emergency response planning, urban planning, new business models.

Data lakes are architectures to support big data ecosystems, and that is why we discuss data lakes next section.

3.2 Data Lake

Data lakes are a recent and trending topic in big data context (Srinivasan and Revathy, 2018; Farid et al., 2016). It is often referred to as an architecture to store big data. James Dixon was the first author to use the data lake concept to refer to a solution to store raw data in a Hadoop ecosystem in 2010 (Dixon, 2010). The first conference paper to cite the term is from 2014, by O'Leary (2014). Data lakes are usually compared to traditional data warehouses, but both concepts differ in several aspects. For instance, unlike data warehouses, data lakes can easily scale and have the ability to store schemaless and multivariate data that are processed just when we need to extract information from the storage system (Cha et al., 2018; Khine and Wang, 2018; Jarke and Quix, 2017; Sreekala PK, 2018; Benaissa et al., 2018). This inherent characteristic makes data lakes quite suitable for big data ecosystems.

Although the concept came up in earlier 2010 (Dixon, 2010), the academia decided to adopt the term only a couple of years later. Thus, there was no consolidated and universally accepted definition, and its functionalities vary according to the context. For example, some say it is only a data repository (Gröger, 2018; Taher et al., 2016; Kathiravelu and Sharma, 2017), while others say it is a complete ecosystem, from data acquisition to information visualization (McPadden et al., 2018; Bhandarkar, 2017; Nogueira et al., 2018). By having these different functionalities, data lakes also present different possible architecture configurations. Regarding data lake architectures, Hadoop is the most commonly used, stand-alone, or in combination with other tools, such as Spark and NoSQL databases.

3.3 Hadoop

Apache Hadoop is an Open-Source Software (OSS) developed in Java programming language and maintained by the Apache Software Foundation[3]. In this study, the concepts we cite refer to Hadoop version 3.X and later. Hadoop started as a sub-project from Apache Nutch in 2005, and the first version was launched in February 2006 (White, 2015). Hadoop is the most used distributed platform for big data storage, processing, and analysis (Forster, 2018; Couto et al., 2019). Hadoop is based on the Google MapReduce paradigm and Google File System, and Douglas Cutting and Mike Cafarella are the co-founders. Yahoo was the first big sponsor of the Hadoop project and still is its primary maintainer. According to the official website (Apache, 2019), the Hadoop Framework is composed of four main modules:

- Common: also known as Hadoop Core. It contains the main features and libraries developed to support the other modules.
- HDFS (Hadoop Distributed File System): a distributed file system that enables storing big data in commodity hardware (based on Google File System (Ghemawat et al., 2003)).
- YARN (Yet Another Resource Negotiator): the framework that manages the cluster resources and job scheduling/monitoring.
- MapReduce: based on Google MapReduce (Dean and Ghemawat, 2004), Hadoop MapReduce enables parallel processing of vast amounts of data on large clusters.

According to Apache (2019), Hadoop HDFS is composed of NameNode and DataNodes, in a master/worker architecture. In non-production environments, we can have only

[3]Available at https://apache.org. Accessed December 2021.

one NameNode in a cluster and some DataNodes - although in real production environments we can have multiple NameNodes. A NameNode is responsible for managing the file system namespace and metadata information and intermediates the clients' access to the files. The DataNodes are responsible for storing the actual data and its replicas. In Hadoop clusters, each file is stored as a sequence of data blocks. Although in Hadoop 2.x, the usual data block size is 128 MB, and the replication factor is 3, these settings are also configurable. The replication factor determines how many times each block is copied and stored in different DataNodes. On Hadoop 3.x, they introduced the HDFS Erasure Coding instead of replication, which decreases the storage overhead by setting the replication factor always to 1, while keeping the same level of fault-tolerance as before. Thereby, Hadoop can assure high availability, even when a node fails.

From the Hadoop ecosystem, Apache Nifi is one of the most relevant tools for data integration. Apache Nifi is a tool to help automate the dataflows among different systems. It is very useful for data ingestion and ETL (Extract, Transform, and Load). Some of the main components of Nifi are the *FlowFile* (objects moving through the system), the *FlowFile Processors* (responsible for performing the tasks), the *Connections* between processors, the *Flow Controller* (also responsible for scheduling), and the *Process Group*, which contains processors and connections (Team, 2021). We can use data profiling techniques to understand the data stored in Hadoop data lakes.

3.4 Data Profiling

According to Abedjan et al. (2017), data profiling is the process of metadata discovery. Abedjan et al. (2015) also states there are three main types of data profile tasks, namely single-column tasks, multi-column tasks, and dependency detection. Single-column tasks are related to domain classification, cardinalities, values distributions, and patterns and data types. Multi-column tasks present correlations and association rules, clusters and outliers, and summaries and sketches. In the dependency detection category, there are three main tasks, namely uniqueness (key discovery, conditional, and approximate), and inclusion dependencies (foreign key discovery, conditional, and approximate), and functional dependencies (conditional and approximate). The *approximate* items are methods that present approximate results for the proposed tasks.

Among the research projects related to data profiling in general, Metanome is well recognized. Metanome (Papenbrock et al., 2015) is a partnership from Hasso-Plattner-Institut with the Qatar Computing Research Institute. Regarding the three main types of data profile tasks, the Metanome project classifies it into nine subgroups, and they make available the implementation for all the algorithms listed in Table 3.1, in the Java programming language.

Table 3.1: Examples of algorithms for data profiling
Source: The author (2022) based on Papenbrock et al. (2015)

Subgroup	Objective	Algorithms
Unique column combination	key discovery	HyUCC and DUCC
Inclusion dependency	foreign-key discovery	BINDER, SPIDER, MANY, FAIDA (approximate)
Functional dependencies	normalization	HyFD, DFD, Tane, Fun, fdep, FastFDs, FdMine, DepMiner, AIDFD (approximate), CFDFinder (conditional)
Multivalued dependencies	normalization	MvDDetector
Order dependencies	data ordering	ORDER
Denial constraints	data cleaning	Hydra, DCFinder
Basic statistics	data exploration	SCDP
Cardinality estimation	zeroth-frequency moment of dataset	FM, PCSA, LC, MAS, BJKST, LogLog, SuperLogLog, MinCount, AKMV, HyperLogLog, Bloom filter, HyperLogLog++
Schema normalization	schema normalization	Normalize (Boyce-Codd normal form)

In other classification, Dai et al. (2016) states there are five primary jobs to be executed by data profiling tools, two in the business domain and 3 in the technical domain. In the technical domain, the main activities are *set profiling, metadata profiling,* and *presentation profiling.* In the business domain, the activities are *content profiling* and *logical rule profiling.* Our work is based on the *set profiling* jobs. Bellow we briefly explain each job:

1. Set profiling: to analyze groups or collections of data, based on statistics, uniqueness, cardinality, frequency, number of rows, distribution, maximum or minimum values, and redundancy.

2. Metadata profiling: related to metadata information discovery (who created the data, what time, data structure).

3. Presentation profiling: process to discover data patterns related to textual and temporal patterns, among others.

4. Content profiling: activity for reviewing basic information about data, such as timeliness, precision, accuracy, null values.

5. Logical rule profiling: data reviewing based on business rules.

Data profiling can benefit people who are working with big data analysis. Taleb et al. (2019) states four benefits of using profiling for data analysis. Firstly, data profiling

helps take correction actions on the data by finding irregularities in data at the beginning of the data processing. Data profiling also helps us access, validate, and analyze metadata easily. Likewise, profiling allows us to perform statistical analysis of data in its source. Finally and most important for our work, data profiling allows us to understand the data source, its structure, content, and data relationships - easing the process of data integration.

3.5 Data integration

Data integration deals with the problem of combining different data sources to provide the user with a unified view (Lenzerini, 2002). There are different approaches for data integration, and some of the most used are based on set similarity. An example of a problem that can be solved with set similarity is the top-k overlap set similarity problem: For all the attributes in all the dataframes, find the top fits for data integration, according to the intersection among the attributes' distinct values (Zhu et al., 2019). The distinct values are one type of set profiling activity, and the intersection among the attributes leads us to an inclusion dependency task: foreign key discovery. We also extensively approach data integration and related background in Chapter 4, Section 4.4.

3.6 Bioinformatics

Bioinformatics is the result of the combination of computer science and biology (Lesk, 2019), where we use software to make inferences about datasets of modern molecular biology, so we can connect the data and extract valuable predictions. There are a lot of bioinformatics datasets available, having the most variate information, formats, types, and size, and it contributes to increase the complexity on jointly analyzing those datasets. We deepen the discussion on the bioinformatics datasets in Chapter 5, Subsection 5.1.2.

Remarks

In summary, in this chapter, we explored the main concepts related to big data integration in Hadoop-based data lakes. We started with the broader idea - big data, then presented how it correlates with the data lake approach, and we pointed out that Hadoop is the most used tool in data lake ecosystems. We also defined data profiling and a correlated task: data integration. Lastly, we presented the concept of bioinformatics since it will be part of our experimental protocol.

4. RELATED WORK

This chapter presents the state-of-the-art studies that we investigate to base our research. We start by broadly mapping the literature about data lakes, and then we narrow our literature search for data profiling in big data. Finally, we deepen our investigation on data integration in data lakes. The main objective of the systematic literature reviews we present hereafter is to find research opportunities to focus our research on.

4.1 Method to perform the literature reviews

This section explains the method we followed for performing the literature reviews. To develop both Mapping Study (MS) and the Systematic Literature Reviews (SLR), we follow the process defined by Brereton et al. (2007). These authors suggest three phases, namely Plan, Conduct, and Document the review, having ten stages among these phases. We also use the Preferred Reporting Items for Systematic Review and Meta-Analysis Protocols (PRISMA-P) (Moher et al., 2015) checklist, which has a set of items that must be addressed to report a systematic review. Next, we describe the stages we followed during the *Planning Phase*

1. Specify Research Question: before we start searching for papers, we must define the research questions. Defining the questions is the most important part of a literature review since it bases all subsequent steps. Aiming to limit and clarify our scope, we followed the PICO (Population, Intervention, Comparison, and Outcome) and PICo (Population, Interest, and Context) methods. These were initially developed by Sackett (2000) to facilitate the elaboration of research definitions. PICO is most used for quantitative papers, while qualitative papers usually apply PICo (Murdoch, 2018). We used PICO and PICo to help us develop the research questions because literature reviews can contain both qualitative and quantitative data.

2. Develop Review Protocol: we developed and applied our search protocol using digital libraries available on the internet. We defined control studies so we could validate our search strings. A control study is a primary study resulting from systematized research, which is known to answer our research questions. We used it to check if the search strings are adequate: if the control study does not show up during string adjustments, we need to readjust the strings until the control study appears among the results.

3. Validate Review Protocol: two researchers developed the review protocols and made several trials changing the search string to obtain results relevant and aligned to

the research question. Then, another senior researcher with a Ph.D. degree in Computer Science validated the protocol. The studies were conducted according to their reviews based on the updated protocol, as presented next.

To *Conduct the Reviews Phase*, we followed the protocol we present hereafter.

1. Identify Relevant Research: we applied the defined search string and, from the results, generated a BibTeX file format for each electronic database. Bibtex is a plain-text file format that contains lists of references, with information about all paper that matches our search criteria.

2. Select Primary papers: in this phase, the researchers must select the papers based on reading only the title, keywords, and abstract of all the papers retrieved from the search engines. To reduce bias, we split the papers to be analyzed between two researchers. We used the Kappa statistic (McHugh, 2012) to measure the level of agreement between the researchers. The Kappa results are based on the number of answers with the same result for both observers (Landis and Koch, 1977). Its maximum value is 1 when the researchers have an almost perfect agreement, and it tends to zero when there is no agreement between them.

   Then, we perform three review rounds, each one containing a random sample of 5% of the population of the paper to be reviewed by the second researcher. For each round, the 2nd researcher received a sample, analyzed each paper, and marked each one as accepted or rejected. Then, we compared the answers: if 1st and 2nd researchers accepted the same paper, we have an agreement in that paper. Then, we calculate the Kappa value for the round. After that, in the papers where there is no agreement, the two main researchers discuss the paper to reach a consensus. If there is still no consensus, we contacted a third researcher to help decide.

   Regarding the SLRs, we also started the selection phase with 1st researcher reviewing and marking each paper as accepted or rejected. Unlike the MS, in the SLR, a second researcher also analyzed all the papers. Finally, we created a spreadsheet that contains information about the selected papers and two columns containing the answer (accepted or rejected) of each researcher for each paper. In the third column, we have information about the agreement for each paper. For example, if 1st and 2nd researchers accepted the same paper, we have an agreement in that paper. However, if there is no agreement, the researchers discuss the paper so they can reach a consensus. If there is still no consensus, we contacted a third researcher to help decide. Based on that process, we calculate the Kappa value.

3. Assess Study Quality: we defined inclusion and exclusion criteria for the papers to retrieve interesting results related to the research topic. To be accepted, papers must meet all the following criteria:

(a) Be qualitative or quantitative research about the theme of interest;
(b) Present a complete study in electronic format;
(c) Be a conference paper or journal.

On the other hand, papers we rejected meet at least one of the following criteria:

(a) Incomplete or short paper (less than four pages);
(b) Unavailable for download;
(c) Do not answer the research questions;
(d) Duplicated study;
(e) Book or book chapter;
(f) Written in another language than English;
(g) Conference proceedings index.

4. Extract Required Data: to help us organize and classify the papers, we used a tool named StArt (State of the Art through Systematic Review)[1]. The StArt is developed by the Federal University of São Carlos, Brazil, and it helps researchers in the process of systematic literature reviews. The StArt has an execution process with 3 phases: papers identification, selection, and extraction. We first register the protocol, and then we register each database and import its bib file, then use StArt to help keep a record of selected papers.

5. Synthesize Data: after finishing all data extraction using StArt, we exported the results to a Google Sheets, so we could analyze and summarize our results. It also helped us to work collaboratively, as Google Sheets is available online.

In the *Document Review Phase*, we finish the reviews by writing and validating the reports.

1. Write Review Report: we use our protocols as a basis to document the reviews after finishing answering the questions of the literature reviews. We present the results for the MS in Section 4.2, for the SLR about big data profiling in Section 4.3, and for the SLR about data integration in data lakes in Section 4.4.

2. Validate Report: once we finished the reports, at least one senior researcher independently reviewed them and suggested improvements that we adjusted to this final version. Reviewers from international conferences also validated our three literature reviews, which we published in the conferences proceedings (Couto et al. (2019, 2022b); Couto and Ruiz (2022).

We first performed an MS based on this method, and then we developed two SLRs. Next, we present the results we achieved with the literature reviews.

[1]Available at http://lapes.dc.ufscar.br/tools/start_tool

4.2 A Mapping Study about Data Lakes

We performed the MS to answer the following research question (RQ): ***What are the definitions and possible big data architectures in data lake ecosystems?*** To better explore the papers, we split the RQ into two, so each accepted paper can answer one or two questions:

- RQ1: What are the most common definitions of the term data lake?
- RQ2: Which system architectures are reported to be used in data lake ecosystems?

We based on PICO and PICo methods to develop the RQ, presented in Table 4.1.

Table 4.1: PICO and PICo definitions for the MS about Data Lakes
Source: Couto et al. (2019)

PICO	PICo
Population: Big data systems **Intervention**: Data lakes **Comparison:** Definition of data lakes **Outcome:** Definition of data lakes and big data architectures in data lakes ecosystems	**Population:** Big data systems **Interest:** Definitions and architectures **Context:** Data lakes

We used the two papers listed in Table 4.2 as control papers. Table 4.3 lists the electronic databases and search strings we used, based on the terms "`data lake*`".

Table 4.2: Control Papers for the MS about Data Lakes
Source: Couto et al. (2019)

Control Study 1	Terrizzano, Ignacio G., et al. "Data Wrangling: The Challenging Journey from the Wild to the Lake." Conference on Innovative Data Systems Research. 2015 (Terrizzano et al., 2015)
Control Study 2	Madera, Cedrine, and Anne Laurent. "The next information architecture evolution: the data lake wave." International Conference on Management of Digital EcoSystems. ACM, 2016 (Madera and Laurent, 2016)

We can see in Table 4.4 the results from the three rounds of analysis of the researcher's agreement. We can see that the level of agreement increased, from moderate in the 1st round to substantial in the 2nd one, and in the last we achieved almost perfect agreement.

It is important to note that we did not set a date range for the search. We found results as early as 1969 referring to data lakes, but upon inspection, we identified that they discussed geological lakes. Thus, given that these do not relate to Computer Science, we discarded them, having the first paper of interest reported in 2014.

MS Results

We started with 662 papers retrieved from the initial search through the web engines. During the process, we identified that 155 papers were duplicated, and we rejected others 419 papers according to the exclusion criteria previously explained. At the end of the MS process, we accepted 87 papers, published between 2014 and 2018. Table 4.5 shows the distribution of papers per database. In this table, we can see that most of the papers came from Springer and Google Scholar. It happens because Springer does not allow us to refine the filter of the studies, so results contain lots of books and book chapters, which we reject, as we explained in the inclusion and exclusion criteria. Google Scholar, in the same way, does not allow complementary filters, frequently redirects to other engines, and it also brings a lot of websites and non-scientific reports among the results.

Among the rejected papers, 75 were published before 2010, when the term data lake was used for the first time in the big data context. We have to manually reject the older ones because, in most electronic databases, we cannot filter results to show only Computer Science related studies. The papers previous to 2010 are mostly from Geology or Civil Engineering. We found that there has been an increasing interest in data lakes, since 2014, with most papers published in 2018.

Another interesting aspect we can see in Table 4.5 is that more than half of the papers we accepted are from Scopus. It happens because Scopus is the largest database of abstracts and scientific citations, compiling more than 71 million records, 23 million titles, and 5,000 publishers, among them ACM, Elsevier, IEEE, Springer, etc. So, we probably accepted papers from other databases using Scopus reference, and then it was marked as duplicated in the original database version.

Table 4.3: Search strings for each electronic database for the MS about Data Lakes
Source: Couto et al. (2019)

Database	Search String
ACM	acmdlTitle:(+data +lake) AND recordAbstract:(+data +lake) AND keywords.author.keyword: (+data +lake)
arXiv	order: -announced_date_first; page_size: 50; primary_classification: cs; terms: AND all="data lake"
Google Scholar	allintitle: "data lake"
IEEE Xplore	((("Document Title":"data lake") OR "Abstract": "data lake") OR "Author Keywords":”data lake”)
Science Direct	Title, abstract, keywords: "data lake"
Scopus	TITLE-ABS-KEY (data lake) AND (LIMIT-TO (SUBJAREA, "COMP")
Springer	https://link.springer.com/search?date-facet-mode=between&facet-language=%22En%22&query=%22data+lake%22&showAll=true&facet-discipline=%22Computer+Science%22
Web of Science	(from all databases): TOPIC: ("data lake") OR TITLE: ("data lake") OR AUTHOR IDENTIFIER: ("data lake")

Table 4.4: Kappa results for the MS about Data Lakes, based on Landis and Koch (1977)
Source: Couto et al. (2019)

Kappa values	Agreement	1° round	2° round	3° round
<0	Poor			
0 – 0.20	Slight			
0,21 – 0,40	Fair			
0,41 – 0,60	Moderate	0.42		
0,61 – 0,80	Substantial		0.64	
0,81 – 1	Almost perfect			0.82

4.2.1 The most common definitions for the term data lake

From the 87 papers we accepted, 71 present data lake definitions. We read each one and copied the definition they present to the term data lake. Then, we created a unique text containing all definitions, and we passed through a web tool to count the words. This tool removed the stopwords, the most frequent terms in English, that are usually removed before natural language processing. Then, it returned a list containing all the other words and the number of times they appear in the text. We analyzed the resulting list and grouped the variances in the same word, by the most frequent one: e.g., *analyses, analyzing,* and *analysis* was grouped into "*analysis*". Table 4.6 shows the top 30 most frequent words. Then, based on the most frequent words, we create a word cloud (Figure 4.1) and a new definition to the term data lake, presented below.

A *data lake is a central repository system for storage, processing, and analysis of raw data, in which the data is kept in its original format and is processed to be queried only when needed. It can store a varied amount of formats in big data ecosystems, from unstructured, semi-structured, to structured data sources.*

Figure 4.1: Word cloud with the terms most commonly related to data lakes
Source: Couto et al. (2019)

Table 4.5: Papers per electronic databases for the MS about Data Lakes
Source: Couto et al. (2019)

Source	Initial	Accepted
ACM	6	0 paper
arXiv	7	1 paper: McPadden et al. (2018)
Google Scholar	197	6 papers: Rajesh and Ramesh (2016); Benaissa et al. (2018); Sreekala PK (2018); Meena and Meena (2016); Terrizzano et al. (2015); Halevy et al. (2016)
IEEE Xplore	32	1 paper: Martínez-Prieto et al. (2017)
Science Direct	19	2 papers: Sharma (2016); Herman et al. (2018)
Scopus	108	53 papers: Bhandarkar (2017); Gollapudi (2015); Diamantini et al. (2018); Gupta et al. (2018); Chen et al. (2017); Revathy and Mukesh (2017); Shlyuger (2017); Ramakrishnan et al. (2017); Golov and Rönnbäck (2017); Miloslavskaya and Tolstoy (2016); Villegas-Ch et al. (2018); Yadav et al. (2015); Farid et al. (2016); Hai et al. (2016); Suriarachchi and Plale (2016a); Maccioni and Torlone (2017); Munshi and Mohamed (2018); Jarke (2017); Dholakia et al. (2017); Brackenbury et al. (2018); O'Leary (2014); Pomp et al. (2018a); Chen et al. (2018); Stefanowski et al. (2017); Pena et al. (2018); Quix et al. (2016); Kondylakis et al. (2018); Dessì et al. (2016); Yamada et al. (2017); Cha et al. (2018); Skluzacek et al. (2016); Wibowo et al. (2017); Fang (2015); Nogueira et al. (2018); Gao et al. (2017); Karpathiotakis et al. (2017); Sankaranarayanan and Lalchandani (2017); Walker and Alrehamy (2015); Suriarachchi and Plale (2016b); Ahmad et al. (2017); Alili et al. (2018); Hai et al. (2018); Rangarajan et al. (2018); Mrozek et al. (2018); Tovernić et al. (2018); Srinivasan and Revathy (2018); Madera and Laurent (2016); Ahmadov et al. (2015); García et al. (2017); Alserafi et al. (2016); Farrugia et al. (2016); Ciociola et al. (2017); Singh et al. (2016)
Springer	222	20 papers: Taher et al. (2016); Kathiravelu and Sharma (2017); Maksymowicz et al. (2016); Taher et al. (2017); Jovanovic et al. (2016); Santos et al. (2016); Ramesh (2015); Le et al. (2018); Solar et al. (2017); Ceravolo et al. (2018); Gröger (2018); McCarthy et al. (2018); Wenning and Kirrane (2018); Dutta (2015); Hui et al. (2018); Jarke and Quix (2017); Kasrin et al. (2018); Alrehamy and Walker (2018); Auer et al. (2017); Kassner et al. (2017)
Web of Science	71	4 papers: Beheshti et al. (2017); Khine and Wang (2018); Mitrovic (2017); Li et al. (2017)

During our analysis, we mapped whom the authors of the papers reference when using a definition for data lakes. We found that James Dixon was the first one to use the term lake in a big data context, in a post in its blog in 2010 Dixon (2010), and he is referenced by ten papers Rajesh and Ramesh (2016); Hai et al. (2016); Khine and Wang

Table 4.6: 30 most frequent words related to data lake definitions
Source: Couto et al. (2019)

Word	Count	Word	Count	Word	Count
data	357	amount	20	scalable	13
lake	105	system	20	schema	13
store	81	big	17	set	13
raw	45	structured	17	structure	13
repository	42	large	16	available	12
formats	37	needed	16	enterprise	12
analysis	29	original	16	Hadoop	12
storage	28	native	14	ingest	12
processed	26	unstructured	14	massive	12
sources	21	various	14	vast	12

(2018); O'Leary (2014); Dessì et al. (2016); Nogueira et al. (2018); Jarke and Quix (2017); Walker and Alrehamy (2015); Alrehamy and Walker (2018); Auer et al. (2017). The first author to reference Dixon's concept in an academic context was O'Leary (2014), in a paper published in 2014. We also discovered the most cited academic definition for data lakes is from Terrizzano et al. (2015), mentioned in twelve papers: Stefanowski et al. (2017); Quix et al. (2016); Martínez-Prieto et al. (2017); Skluzacek et al. (2016); Wibowo et al. (2017); Halevy et al. (2016); Gao et al. (2017); Walker and Alrehamy (2015); Suriarachchi and Plale (2016b); Hai et al. (2018); García et al. (2017); Alserafi et al. (2016).

4.2.2 System architectures reported to be used in data lakes ecosystems

System architectures are formed by a set of tools that work together to achieve the environmental objective. Among the 87 papers we mapped, we identified 117 different tools used in data lake architectures. Of all the tools, Apache Hadoop was the most mentioned, having 37 citations (Ahmad et al., 2017; Gollapudi, 2015; Ramesh, 2015; Fang, 2015; Taher et al., 2016; Rajesh and Ramesh, 2016; Farid et al., 2016; Suriarachchi and Plale, 2016b; Madera and Laurent, 2016; Farrugia et al., 2016; Kathiravelu and Sharma, 2017; Taher et al., 2017; Bhandarkar, 2017; Ramakrishnan et al., 2017; Suriarachchi and Plale, 2016a; Stefanowski et al., 2017; Shlyuger, 2017; Karpathiotakis et al., 2017; Auer et al., 2017; Martínez-Prieto et al., 2017; Sankaranarayanan and Lalchandani, 2017; Li et al., 2017; Kassner et al., 2017; García et al., 2017; McPadden et al., 2018; Gupta et al., 2018; Revathy and Mukesh, 2017; Le et al., 2018; Ceravolo et al., 2018; Gröger, 2018; Benaissa et al., 2018; Sreekala PK, 2018; Munshi and Mohamed, 2018; Khine and Wang,

Table 4.7: Architectures: the most used tools in data lakes
Source: Couto et al. (2019)

Tool	Amount	Papers
1) Ingestion		
Apache Kafka	10	Taher et al. (2017); Bhandarkar (2017); Shlyuger (2017); Ramakrishnan et al. (2017); Stefanowski et al. (2017); Ahmad et al. (2017); Auer et al. (2017); McPadden et al. (2018); Gupta et al. (2018); Le et al. (2018)
Apache Flume	7	Suriarachchi and Plale (2016a); Martínez-Prieto et al. (2017); Ahmad et al. (2017); Auer et al. (2017); García et al. (2017); Munshi and Mohamed (2018); Rangarajan et al. (2018)
Apache Sqoop	5	Martínez-Prieto et al. (2017); Ahmad et al. (2017); García et al. (2017); McPadden et al. (2018); Le et al. (2018)
Apache Nifi	3	Shlyuger (2017); McPadden et al. (2018); Ahmad et al. (2017)
Komadu	2	Suriarachchi and Plale (2016b,a)
Talend Studio	2	Ahmad et al. (2017); Wibowo et al. (2017)
2) Storage		
Apache Cassandra	6	Kondylakis et al. (2018); Karpathiotakis et al. (2017); Auer et al. (2017); Kasrin et al. (2018); Ahmad et al. (2017); Dutta (2015)
MongoDB	6	Ciociola et al. (2017); Hai et al. (2018); Herman et al. (2018); Kathiravelu and Sharma (2017); Nogueira et al. (2018); Kasrin et al. (2018)
Apache HBase	4	Gupta et al. (2018); Ramesh (2015); Ahmad et al. (2017); Le et al. (2018)
MySQL	4	Hai et al. (2018); Kathiravelu and Sharma (2017); Suriarachchi and Plale (2016b); Ahmad et al. (2017)
Neo4J	3	Pomp et al. (2018a); Taher et al. (2017); Walker and Alrehamy (2015)
Oracle	3	Kassner et al. (2017); Ceravolo et al. (2018); Ahmad et al. (2017)
Apache Mahout	2	Kassner et al. (2017); Ahmad et al. (2017)
GlusterFS	2	Li et al. (2017); Pena et al. (2018)
PostgreSQL	2	Kasrin et al. (2018); Skluzacek et al. (2016)
3) Processing		
Apache Spark	26	Maksymowicz et al. (2016); Farid et al. (2016); Suriarachchi and Plale (2016b); Bhandarkar (2017); Ramakrishnan et al. (2017); Suriarachchi and Plale (2016a); Maccioni and Torlone (2017); Stefanowski et al. (2017); Martínez-Prieto et al. (2017); Karpathiotakis et al. (2017); Ahmad et al. (2017); Auer et al. (2017); Kassner et al. (2017); García et al. (2017); McPadden et al. (2018); Revathy and Mukesh (2017); Le et al. (2018); Ceravolo et al. (2018); Benaissa et al. (2018); Munshi and Mohamed (2018); Khine and Wang (2018); Dholakia et al. (2017); Hai et al. (2018); Herman et al. (2018); Ramesh (2015); Wenning and Kirrane (2018)
Apache Hive	11	Ramesh (2015); Farrugia et al. (2016); Kathiravelu and Sharma (2017); Bhandarkar (2017); Ahmad et al. (2017); Auer et al. (2017); García et al. (2017); McPadden et al. (2018); Gupta et al. (2018); Ceravolo et al. (2018); Munshi and Mohamed (2018)
Apache Storm	7	Suriarachchi and Plale (2016b,a); Stefanowski et al. (2017); Ahmad et al. (2017); Kassner et al. (2017); McPadden et al. (2018); Pomp et al. (2018a)
Apache Impala	4	Ramesh (2015); Ceravolo et al. (2018); Munshi and Mohamed (2018); Tovernić et al. (2018)
Apache Drill	4	Revathy and Mukesh (2017); Pomp et al. (2018a); Kathiravelu and Sharma (2017); Ceravolo et al. (2018)
Apache Oozie	4	García et al. (2017); McPadden et al. (2018); Martínez-Prieto et al. (2017); Ahmad et al. (2017)
Python	4	Ciociola et al. (2017); Herman et al. (2018); Tovernić et al. (2018); Ahmad et al. (2017)
Apache Flink	3	Stefanowski et al. (2017); Khine and Wang (2018); Auer et al. (2017)
Apache Pig	3	Martínez-Prieto et al. (2017); Ahmad et al. (2017); García et al. (2017)
Apache POI	2	Ahmad et al. (2017); Quix et al. (2016)
Kepler	2	Suriarachchi and Plale (2016b,a)
Shiny	2	García et al. (2017); Martínez-Prieto et al. (2017)
Splunk	2	Ramesh (2015); Ahmad et al. (2017)
WEKA	2	Kassner et al. (2017); Wibowo et al. (2017)
4) Presentantion		
Microsoft Power BI	2	Tovernić et al. (2018); Villegas-Ch et al. (2018)
Tableau	2	Munshi and Mohamed (2018); Ahmad et al. (2017)
5) Security		
Apache Ranger	4	Revathy and Mukesh (2017); Gupta et al. (2018); McPadden et al. (2018); Rangarajan et al. (2018)
Kerberos	3	McPadden et al. (2018); Rangarajan et al. (2018); Revathy and Mukesh (2017)
Apache Ambari	2	Revathy and Mukesh (2017); McPadden et al. (2018)
Apache Knox	2	Revathy and Mukesh (2017); Gupta et al. (2018)
Apache Sentry	2	Revathy and Mukesh (2017); Gupta et al. (2018)

2018; Dholakia et al., 2017; Pomp et al., 2018a; Rangarajan et al., 2018; Tovernić et al., 2018). Hadoop is used as basis for the other tools. Table 4.7 lists the tools cited more than once. We list and categorize the remaining tools in five groups, described as follows.

1) Ingestion

Class of tools that work on data acquisition and collection, from the most varied sources. In this group, the most cited tool is Apache Kafka, which consists of a high-capacity, low latency distributed streaming platform for real-time data processing.

2) Storage

Represented by tools to store, integrate, and normalize data. Besides Hadoop, Apache Cassandra and MongoDB, are the most popular for this group. Apache Cassandra is a highly scalable, column-driven distributed database that has a data model based on Google's BigTable. MongoDB is a multi-platform, document-oriented database that stores data in JSON documents with the dynamic schema. It can be considered the most famous NoSQL database on the market.

3) Processing

Tools in this group are responsible for analyzing, processing, and transforming the raw data so that we can extract information from it. In this group, Apache Spark is the most cited in all papers, besides Apache Hadoop. It is a framework for distributed computing that provides an interface for clustered programming with parallelism and fault tolerance.

4) Presentation

Tools that help us make sense of the data in a visual manner. In this case, Microsoft Power BI and Tableau are the most mentioned in the papers. Power BI is a cloud-based Business Analytics service focused on presenting information across dashboards. Tableau is a software for interactive data visualization.

5) Security

Includes tools to manage system authentication and authorization, assure data security, permit auditing, and allow data encryption. Apache Ranger is the most mentioned. It is a framework for activating, monitoring, and managing data security in the Apache Hadoop ecosystem. According to our analysis, the Apache Software Foundation (ASF) develops most of the tools reported in the studies for data lake architectures, helping to create the most used ecosystems.

4.2.3 Conclusions of the MS about data lakes

This section presented a systematic mapping study to explain data lake definition and architecture. We started with 662 papers, and we end up with 87 in the final set, after our criteria selection. We selected papers from 2014 to 2018 and came from eight different electronic databases. We learned that the term data lake was first used in 2010 to designate a big data system. We proposed a new definition from the selected papers in our study for the concept data lake. We also found that Hadoop and its ecosystem comprise the most frequent architecture to build data lakes. One limitation of our study is that we choose to limit the search only to the papers that have the term "data lake". We know that many researchers can be working with data lakes without using this buzzword, but as we want to know its definition, we chose to accept that limitation.

4.3 A Systematic Literature Review about Big Data Profiling

After performing the MS about data lakes, we noticed a gap in the literature related to big data profiling architectures. Then, we decided to deeper explore this topic. So, we decided to perform an SLR to increase the probability that we reach the most relevant literature related to big data profiling characterization. In this section, we present the protocol and the results we achieved. We execute the SLR based on the method presented in Section 2. Hereafter we answer the following research questions:

- RQ1: What are the most used tools for big data profiling?
- RQ2: What are the scenarios and datasets reported to be profiled?
- RQ3: What type of metadata did the papers collect?
- RQ4: Which information is extracted using data profiling?
- RQ5: What are the challenges in big data profiling?

In the same way we did in the mapping study we previously performed, we used PICO and PICo methods (see Table 4.8) to help us develop the RQs. Table 4.9 shows the paper we use as control for the search strings.

We performed the search for the terms `"data profiling" AND ("big data" OR "data lake")` through eight different electronic databases. Table 4.10 lists search strings we used to for each database. For ArXiv and ACM, we joined two search strings because the results obtained using both are more adherent to the research topic.

Table 4.8: PICO and PICo definitions for the SLR about data profiling in big data
Source: Couto et al. (2022b)

PICO	PICo
Population: Big data systems **Intervention**: Data profiling **Comparison:** Traditional data warehouses **Outcome:** Tools, data types, scenarios, types of metadata, and challenges	**Population:** Big data systems **Interest:** Tools, data types, scenarios, types of metadata, and challenges **Context:** Data profiling

Table 4.9: Control Paper for the SLR about data profiling in big data
Source: Couto et al. (2022b)

Control Study 01	Juddoo, Suraj. "Overview of data quality challenges in the context of Big Data." International Conference on Computing, Communication, and Security. IEEE, 2015 Juddoo (2015).

We also used the Kappa method to measure the agreement among researchers. Kappa results are presented in Table 4.11. We only performed one review round, where both researchers analyzed all the papers. We can see in Table 4.11 that we achieved almost perfect agreement. It happened because the researchers discussed the main objectives before starting the SLR, so they were aligned.

SLR Results

This section presents our analysis for the papers, and we answer our five RQ. From the 103 initial set of papers, we accepted 20 papers following the SLR process. Table 4.12 shows the number of papers we retrieved and the ones we accepted per database. In this Table, we can see that most of the papers came from Scopus and IEEE Xplore.

The papers we accepted were published between 2013 to 2019, being 2018 and 2019 the years with most of the papers (Figure 4.2b). It demonstrates the growing interest of researchers on this topic over the years. Also, we analyze publications by country of publication using the first author's institutional affiliation. The review included papers from eleven countries. Figure 4.2a shows that most papers are from researchers from the United States (4 papers) and Germany (4 papers).

Table 4.10: Search strings for each electronic database for the SLR about data profiling in big data

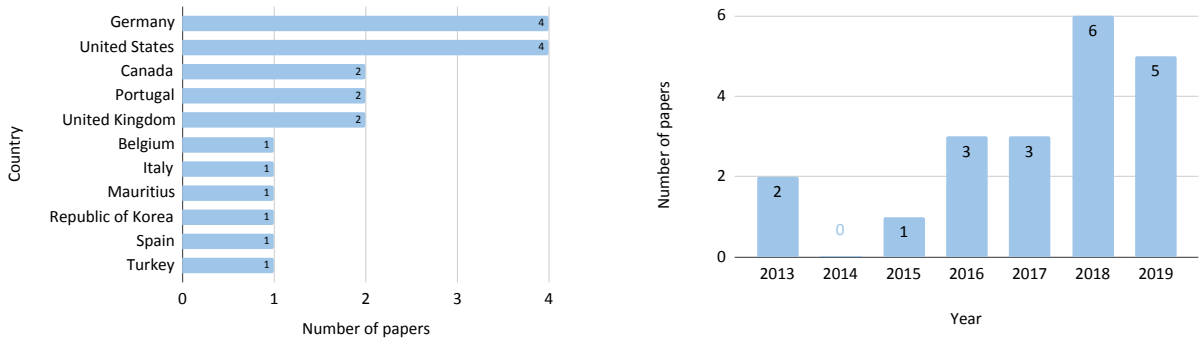
Source: Couto et al. (2022b)

Electronic Database	Search String
ACM	(Searched for acmdlTitle:(+"data profiling" +"big data") OR recordAbstract:(+"data profiling" +"big data") OR keywords.author.keyword:(+"data profiling" +"big data")) *JOIN* (Searched for acmdlTitle:(+"data profiling" +"data lake") OR recordAbstract:(+"data profiling" +"data lake") OR keywords.author.keyword:(+"data profiling" +"data lake"))
arXiv	(Query: order: -announced_date_first; size: 50; include_cross_list: True; terms: AND all="data profiling"; AND all="big data") *JOIN* (Query: order: -announced_date_first; size: 50; include_cross_list: True; terms: AND all="data profiling"; AND all="data lake")
Google Scholar	allintitle: "big data" OR "data lake" "data profiling"
IEEE Xplore	("All Metadata":"data profiling" AND ("big data" OR "data lake"))
Science Direct	Title, abstract, keywords: "data profiling" AND ("big data" OR "data lake")
Scopus	(TITLE-ABS-KEY ("data profil*") AND TITLE-ABS-KEY ("big data" OR "data lake*"))
Springer	https://link.springer.com/search?dc.title=%22data+profiling%22+%28%22big+data%22+OR+%22data+lake%22%29&date-facet-mode=between&showAll=true
Web of Science	(from all databases): TOPIC: ("data profiling") AND TOPIC: ("big data" OR "data lake") Timespan: All years. Databases: WOS, DIIDW, KJD, RSCI, SCIELO. Search language=Auto

Table 4.11: Kappa results for the SLR about data profiling in big data, based on Landis and Koch (1977)

Source: Couto et al. (2022b)

Kappa values	Strength of agreement	Value
<0	Poor	
0 – 0.20	Slight	
0,21 – 0,40	Fair	
0,41 – 0,60	Moderate	
0,61 – 0,80	Substantial	
0,81 – 1	Almost perfect	0.84



(a) Publications per country.

(b) Publications per year.

Figure 4.2: Number of publications per country and year
Source: Couto et al. (2022b)

Summary of the accepted papers

This section presents an outline regarding the approaches reported in the papers. Among the papers, four of them reviewed the literature about data profiling tasks and tools

Table 4.12: Papers per electronic databases for the SLR about data profiling in big data
Source: Couto et al. (2022b)

Electronic Database	Initial	Accepted papers
ACM	5	2 papers: Maccioni and Torlone (2017); Shaabani and Meinel (2018)
arXiv	2	0 paper
Google Scholar	17	1 paper: Alserafi et al. (2016)
IEEE Xplore	11	5 papers: Abedjan et al. (2016); Koehler et al. (2019); Liu et al. (2013); Canbek et al. (2018); Juddoo (2015)
Science Direct	3	2 papers: Ardagna et al. (2018); Vieira et al. (2020)
Scopus	38	7 papers: Jang et al. (2018); Taleb et al. (2019); Sun et al. (2018); Santos et al. (2019); Heise et al. (2013); Chrimes and Zamani (2017); Khalid and Zimányi (2019)
Springer	8	2 papers: Abedjan (2018); Dai et al. (2016)
Web of Science	19	1 papers: Sampaio et al. (2019)

classification (Abedjan et al., 2016; Abedjan, 2018; Dai et al., 2016; Juddoo, 2015); two papers are about general classification (Vieira et al., 2020; Sun et al., 2018); two papers are about data wrangling (Sampaio et al., 2019; Koehler et al., 2019); three are about data lakes Alserafi et al. (2016); Maccioni and Torlone (2017); Khalid and Zimányi (2019); five papers are about data quality (Ardagna et al., 2018; Taleb et al., 2019; Chrimes and Zamani, 2017; Juddoo, 2015; Jang et al., 2018); and five present varied approaches (Liu et al., 2013; Heise et al., 2013; Shaabani and Meinel, 2018; Canbek et al., 2018; Santos et al., 2019).

The authors Abedjan et al. (2016); Abedjan (2018); Dai et al. (2016) reviewed the literature for classifying data profiling tasks and tools. In Dai et al. (2016), they review the literature about data profiling, and then they propose a new definition and classification for data profiling tasks and existing tools. Then, they present data quality metrics and score calculations and present a framework for data profiling in big data. In Abedjan et al. (2016) and Abedjan (2018), they classify data profiling tasks and review the state-of-the-art about data profiling systems and techniques. Unlike our work, they do not follow a structured method to perform systematic reviews, and, besides the tools classification, we answer different research questions, including scenarios, datasets, metadata, information extracted, and the main challenges related to big data profiling.

Also, Vieira et al. (2020) and Sun et al. (2018) perform classification tasks, however Vieira et al. (2020) use classification to quantify the impact in the volume of data used and Sun et al. (2018) use classification for data prediction.

Regarding data wrangling, the authors Sampaio et al. (2019) develop a conceptual approach based on a domain-specific language for data wrangling, which is a process for data quality improvement that includes data profiling. In this case, they use data profil-

ing to identify quality issues. In its turn, Koehler et al. (2019) presents a wrangling process to use data context for data wrangling automation.

Papers Ardagna et al. (2018); Taleb et al. (2019); Chrimes and Zamani (2017); Juddoo (2015); Jang et al. (2018) address data quality. Ardagna et al. (2018) propose a data quality service to analyze big data. They present a methodology to help the user tune the parameters to best fit its intentions: budget minimization, time minimization, or confidence maximization. To do so, they create a model named CCT (Confidence/Cost/-Time) that captures the interrelationships between non-functional requirements. Taleb et al. (2019) propose a Big Data Quality Profiling Model that involves several modules such as sampling, profiling, exploratory quality profiling, quality profile repository, and data quality profile. Jang et al. (2018) propose a data profiling model using statistical analysis techniques to derive attributes for big data quality diagnosis. Chrimes and Zamani (2017) establish an interactive big data analytics platform with simulated patient data. They used open-source software technologies that were achieved by constructing a platform framework with HDFS using HBase (a key-value NoSQL database). Furthermore, Juddoo (2015) also presents a literature review, focusing on an overview of data quality challenges in the context of Big Data.

Some papers also report outcomes related to data lake (Alserafi et al., 2016; Maccioni and Torlone, 2017; Khalid and Zimányi, 2019). Alserafi et al. (2016) present a framework for data governance in data lakes. They propose techniques for the automatized analytical discovery of cross-data lake content relationships (information profiles). Thus, they perform metadata annotation, extraction, management, and exploitation, to identify duplicate datasets, relations among datasets, and outlier datasets. Further, Maccioni and Torlone (2017) propose Kayak to expedite data preparation in a data lake. Kayak is a data management framework that implements adhoc primitives and executes them with an efficient strategy. Khalid and Zimányi (2019) developed goal-based and rule-based agents to generate metadata profiles in a data lake. The rule-based agent operates on rules to categorize metadata files, and the goal-based agents work on goals to differentiate the types of metadata and add sections to the metadata profiles.

Among the papers that present varied approaches, Liu et al. (2013) present an integrated method to address the heterogeneity issue in modeling big time-series data, while Heise et al. (2013) presented a strategy to find all the unique and non-unique combinations of columns in a dataset; Shaabani and Meinel (2018) propose a system for inclusion dependency injection discovery; Canbek et al. (2018) created a profiling approach to gain insights about a group of datasets in different dimensions, using four data profiling techniques, namely basic profiling, timeline profiling, duplicate samples profiling, and Density/sparsity profiling. Finally, Santos et al. (2019) present a study about big data Warehousing, state data profiling is a part of BDW entities resolution, which is a component that addresses the integration of data and business processes in a BDW.

Now we will present the answers to the research questions, using the analysis we performed about the papers we have just briefly described.

4.3.1 The most used tools for big data profiling

We found 15 papers that mention the tool they use to generate data profiling in big data. The paper that presents most tools are Dai et al. (2016) (nine tools). We also found six papers that present tools developed by the authors (Liu et al., 2013; Heise et al., 2013; Alserafi et al., 2016; Shaabani and Meinel, 2018; Khalid and Zimányi, 2019; Maccioni and Torlone, 2017). Among the most cited tools, we found R and Python programming languages, and Talend, briefly described as follows.

- R programming language[2]: reported by 4 papers. (Jang et al., 2018; Canbek et al., 2018; Taleb et al., 2019; Sampaio et al., 2019). R is a free software environment, mainly composed of the programming language and often used with RStudio IDE[3]. R is mostly used for statistical computing and graphics generation.

- Python programming language[4]: presented in 3 papers (Canbek et al., 2018; Taleb et al., 2019; Khalid and Zimányi, 2019). According to StackOverflow (2019), Python is the language most people want to work with and is among the most loved by the developers. Python is also used for statistical computing and often for developing machine learning models.

- Talend[5]: cited by three papers (Vieira et al., 2020; Sampaio et al., 2019; Dai et al., 2016). Talend is an open-source tool for data integration that provides data collection, government, transformation, and sharing services.

Other tools are also mentioned once in the papers. Some have a free version, and others are commercial. Among the tools that presents at least one free version, we found DataCleaner tool, Aggregate Profiler Tool, and Talend Open Studio for Data quality (Dai et al., 2016), Hadoop, Kafka, and Zookeeper (Heise et al., 2013),OpenRefine and Apache Taverna (Sampaio et al., 2019), Apache Spark (Taleb et al., 2019), MongoDB (Canbek et al., 2018), and HBase (Chrimes and Zamani, 2017).

As for the commercial tools, papers reported using Informatic Data Profiling, SAP Information Steward, Oracle Enterprise Data Quality, Collibra Data Stewardship Manage, IBM InfosSphere Information Analyzer, and SAS DataFlux Data Management Studio (Dai et al., 2016), and Trifacta Wrangler (Sampaio et al., 2019).

[2]Available at https://www.r-project.org Accessed in November, 2019.
[3]Available at https://rstudio.com Accessed in November, 2019.
[4]Available at https://www.python.org Accessed in November, 2019.
[5]Available at https://www.talend.com Accessed in November, 2019.

Following the results we obtained on the MS, we noticed that some tools such as Hadoop, Kafka, Zookeeper, Hive, and Spark, often related to the Hadoop ecosystem, also are used as a basis for big data profiling.

4.3.2 Scenarios and datasets reported to be profiled

We identified 13 papers that describe the scenarios where they perform big data profiling. Among these, 4 presented more than one scenario (Alserafi et al., 2016; Shaabani and Meinel, 2018; Vieira et al., 2020; Koehler et al., 2019). We group these scenarios into seven categories as follows, and we describe the corresponding datasets:

- Automotive: taxi and Uber (Sun et al., 2018), urban traffic (Sampaio et al., 2019), cars, and crashes datasets (Alserafi et al., 2016).
- Business: papers that used stock and strikes datasets (Alserafi et al., 2016), a business decision-support database benchmark (TPC-H) (Heise et al., 2013), and financial data (Koehler et al., 2019).
- City: data related to smart city (Ardagna et al., 2018), weather (Jang et al., 2018), and road safety (Khalid and Zimányi, 2019), real-estate domain, and the United Kingdom open government data portal (Koehler et al., 2019).
- Health: includes papers that used breast cancer and diabetes datasets (Alserafi et al., 2016), hospital data (Chrimes and Zamani, 2017), and biological databases (H-Genome, Mb, Pdb) (Shaabani and Meinel, 2018).
- Industry: data related to power plant (Liu et al., 2013), supply chain, and automotive electronics industry (Vieira et al., 2020).
- Web: includes Wikipedia data, linked open data about famous persons, anonymized web-log data, and open music encyclopedia data (Shaabani and Meinel, 2018).
- Others: includes papers that used basketball, tae, and tic-tac-toe games, a dataset about flowers (Iris) (Alserafi et al., 2016), and an Android mobile malware dataset (Canbek et al., 2018).

The papers present three types of datasets: online repositories (Alserafi et al., 2016; Jang et al., 2018; Khalid and Zimányi, 2019; Shaabani and Meinel, 2018), real-world datasets generated by the researchers (Liu et al., 2013; Canbek et al., 2018; Ardagna et al., 2018; Sun et al., 2018; Sampaio et al., 2019; Vieira et al., 2020; Koehler et al., 2019), and generated data (Chrimes and Zamani, 2017; Heise et al., 2013). Online repositories include OpenML, Kaggle, biological databases, and Wikipedia. The authors who generated

their datasets used data from medical records, GPS sensors, fare collection systems that collect data, and open government data portals, for example. Finally, Heise et al. (2013) also uses a database benchmark to generate data.

4.3.3 Type of metadata collected by the papers

We found four studies that reported the metadata they collected. Koehler et al. (2019) presented structural properties, column name tokens, column names, data types, schema paths, and parent and leaf relationships in the schema. Khalid and Zimányi (2019) reported the collection of column count, data types, number of rows, data domain, date of data publishing, dataset origin, labeling definitions, data previews, column descriptions, creation date, data labels, data variables, attribute counter, attribute lists, number of missing data values, version management, metadata identifier, and metadata type. Shaabani and Meinel (2018) collected dataset name, size, number of non-empty relations (tables), number of attributes (columns), number of tuples (rows), minimum, maximum, and average number of tuples per relation, and number of unary inclusion dependencies in the dataset. Chrimes and Zamani (2017) stated that the metadata they collected is: admin source, admin type, and encounter type (the kind of attendance in a hospital).

Most papers do not report the type of metadata they collect to generate data profiling since most use data profiling to create metadata, unlike the above papers that used metadata to create data profiling. Thus, next, we will verify that all accepted papers answer RQ4, about which information was generated from the big data profiling.

4.3.4 Information extracted using data profiling

Based on the analysis of the selected papers, we create a classification with the nine types of information most commonly extracted using big data profiling. Table 4.13 shows a summary, with the number of papers per type of information. Bellow, we detail each class and present the type of information extracted for each paper, sorted by the number of papers in descending order.

- Statistics: papers in this group reported using data profiling for presenting a numerical analysis about the data. The papers mentioned data correlation and association rules (Abedjan, 2018; Alserafi et al., 2016; Dai et al., 2016), check data distribution (Canbek et al., 2018; Abedjan, 2018; Abedjan et al., 2016; Khalid and Zimányi, 2019; Santos et al., 2019; Dai et al., 2016), check data cardinality, or number of distinct values (Abedjan, 2018; Canbek et al., 2018; Dai et al., 2016; Heise et al., 2013; Abedjan et al., 2016), check the number of null values (Abedjan et al., 2016; Juddoo, 2015; Dai

et al., 2016; Ardagna et al., 2018), check mean, standard deviations, minimum and maximum values (Dai et al., 2016; Ardagna et al., 2018; Alserafi et al., 2016), missing values (Jang et al., 2018), and general data summaries (Abedjan, 2018; Abedjan et al., 2016; Khalid and Zimányi, 2019; Koehler et al., 2019; Taleb et al., 2019).

- Dependencies: papers in this group presented the use of data profiling for finding relationships between different datasets, or data attributes, or columns in the same dataset, such as discovering foreign keys (Shaabani and Meinel, 2018; Juddoo, 2015; Dai et al., 2016), detecting functional dependencies and inclusion dependencies (Abedjan, 2018; Abedjan et al., 2016; Maccioni and Torlone, 2017; Koehler et al., 2019; Chrimes and Zamani, 2017; Heise et al., 2013), and compute joinability and affinity between two datasets (Maccioni and Torlone, 2017).
- Quality: this group of papers reported using data profiling to discover data issues (Sampaio et al., 2019; Santos et al., 2019), such as outliers (Abedjan, 2018; Jang et al., 2018), syntactic errors (Vieira et al., 2020), and data quality details, such as missing data and data problems (Taleb et al., 2019).
- Data characteristics: these papers reported presenting basic profiling (Canbek et al., 2018; Maccioni and Torlone, 2017), data structures and data creator (Dai et al., 2016), descriptive information about data sources (Koehler et al., 2019), and data characteristics, character lengths, and data sources (Chrimes and Zamani, 2017).
- Data classification: in this group, they perform data categorization and clusterization, creating groups according to the data profile (Abedjan, 2018; Khalid and Zimányi, 2019; Sun et al., 2018).
- Data patterns: these papers reported using data profiling to help find interesting data patterns and behavior (Abedjan, 2018; Abedjan et al., 2016; Dai et al., 2016).
- Timeliness: they present temporal data, using profiling techniques: age and freshness of the dataset (Canbek et al., 2018), time of creation and time patterns (Dai et al., 2016), and the trajectory of feature values along the time (Liu et al., 2013).
- Business processes and rules: they use data profiling to understand business rules (Dai et al., 2016) and business processes (Santos et al., 2019).

4.3.5 Challenges in big data profiling

During our analysis, we found ten papers that report 15 challenges the authors face when performing big data profiling. We list the challenges alongside its descriptions as follows.

Table 4.13: Number of papers per type of information extracted using data profiling in big data

Source: Couto et al. (2022b)

#	Type of information	Papers
14	statistics	Abedjan (2018); Alserafi et al. (2016); Canbek et al. (2018); Dai et al. (2016); Sampaio et al. (2019); Taleb et al. (2019); Abedjan et al. (2016); Ardagna et al. (2018); Heise et al. (2013); Jang et al. (2018); Juddoo (2015); Khalid and Zimányi (2019); Maccioni and Torlone (2017); Santos et al. (2019)
9	dependencies	Abedjan (2018); Abedjan et al. (2016); Chrimes and Zamani (2017); Dai et al. (2016); Heise et al. (2013); Juddoo (2015); Koehler et al. (2019); Maccioni and Torlone (2017); Shaabani and Meinel (2018)
6	quality	Abedjan (2018); Jang et al. (2018); Sampaio et al. (2019); Santos et al. (2019); Taleb et al. (2019); Vieira et al. (2020)
5	data characteristics	Canbek et al. (2018); Chrimes and Zamani (2017); Dai et al. (2016); Koehler et al. (2019); Maccioni and Torlone (2017)
3	data classification	Abedjan (2018); Khalid and Zimányi (2019); Sun et al. (2018)
3	data patterns	Abedjan (2018); Abedjan et al. (2016); Dai et al. (2016)
3	timeliness	Canbek et al. (2018); Dai et al. (2016); Liu et al. (2013)
2	business processes and rules	Dai et al. (2016); Santos et al. (2019)

1. Complexity: data profiling is a complex operation that belongs to the data preparation process (Maccioni and Torlone, 2017). Variety and volume create challenges related to computational complexity, such as memory requirements (Juddoo, 2015). Complexity related to the environment increases the challenges in big data profiling (Abedjan, 2018; Abedjan et al., 2016; Canbek et al., 2018; Dai et al., 2016).

2. Continuous profiling: automatically updating data profiling on the fly, while more data is entering the system is challenging because it requires the data profiling algorithms to be always running, and it spends resources that could be used for other tasks (Juddoo, 2015).

3. Incremental profiling: updating data profiling according to a predetermined amount of time (Juddoo, 2015).

4. Interpretation: being able to understand and interpret data profiling results (Taleb et al., 2019; Abedjan, 2018; Abedjan et al., 2016; Canbek et al., 2018; Juddoo, 2015).

5. Lack of research: the authors state there is not much research on big data profiling research topic (Taleb et al., 2019).

6. Metadata: creating metadata is the biggest challenge (Khalid and Zimányi, 2019), since metadata can be created manually or through data profiling. Selecting the proper metadata to generate data profiling is another challenge (Abedjan et al., 2016; Abedjan, 2018).

7. Online profiling: present intermediate results to the user, with a predefined confidence level (Juddoo, 2015).

8. Poor data quality: data quality impacts on data profiling results veracity (Abedjan et al., 2016; Abedjan, 2018; Ardagna et al., 2018; Dai et al., 2016; Juddoo, 2015; Taleb et al., 2019).

9. Profiling dynamic data: profiling dynamic data, such as streams, is an open challenge because these types of data change often, and it makes previous profiling obsolete (Abedjan et al., 2016; Dai et al., 2016).

10. Topical profiling: data profiling traditional techniques usually do not consider the whole context, such as data semantics (Vieira et al., 2020), data values, structures and standards, business rules, and characteristics (Jang et al., 2018), or the use of specific datasets, such as social media (Juddoo, 2015).

11. Value: it is challenging to use data profiling for transforming the proper data in decision support so that we can generate value (Ardagna et al., 2018; Canbek et al., 2018; Juddoo, 2015).

12. Variability: data that vary regarding size, content, and other aspects, and it requires the use of different algorithms at the same time to be able to profile different data (Liu et al., 2013; Alserafi et al., 2016; Canbek et al., 2018)

13. Variety: profiling of heterogeneous data (audio, text, video, etc.) (Abedjan et al., 2016; Alserafi et al., 2016; Canbek et al., 2018; Dai et al., 2016; Juddoo, 2015; Vieira et al., 2020).

14. Visualization: generate visualizations to help understand data profiling (Abedjan et al., 2016; Taleb et al., 2019).

15. Volume: challenges related to the size of the datasets (Juddoo, 2015; Abedjan et al., 2016; Canbek et al., 2018; Dai et al., 2016).

4.3.6 Conclusions of the SLR about data profiling

We performed a systematic literature review to characterize big data profiling, mapping the tools, scenarios, datasets, metadata, information, and related challenges. By

applying the SLR process, we selected 20 papers that answer at least one of our research questions, published from 2013 to 2019. Thus, we conclude that big data profiling is a reasonably new research topic. During papers analysis, we found that R and Python programming languages are among the most used tools for big data profiling, alongside the Hadoop ecosystem and other commercial tools.

We also classified the scenarios presented in the papers into seven groups: automotive, business, city, health, industry, web, and others. We also mapped the datasets reported in the papers. When we searched for the metadata the papers reported using, we found only four papers that followed the approach of using metadata to create data profiling. On the other hand, all the accepted papers presented the information they extracted using data profiling. We group the information into nine types: statistics, dependencies, quality, data characteristics, data classification, data patterns, timeliness, and business processes and rules. Most of the papers use data profiling for presenting statistics (70%), dependencies (45%), and information about data quality (30%).

Finally, we map and describe the challenges reported by the authors when performing big data profiling. We identified 15 challenges, which we consider when developing our management approach for Hadoop-based data lakes.

4.4 A Systematic Literature Review about Data Integration in data lakes

To conclude the literature reviews, we developed a last SLR to deepen the knowledge about data integration in data lakes. In this section, we present the protocol and the results we achieved. We execute the SLR based on the method presented in Section 2. Hereafter we answer the following research questions:

- RQ1: What are the models for data integration in data lakes?
- RQ2: Which similarity metrics are used for data integration?
- RQ3: How do they evaluate data integration models for data lakes?
- RQ4: What type of data do they integrate?
- RQ5: What are the challenges in data integration in data lakes?

In the same way, we did in the mapping study we previously performed, we used PICO and PICo methods (see Table 4.14) to help us develop the RQs. Table 4.15 shows the paper we use as a control for the search strings.

We performed the search for the terms `"data integration" AND "data lake"` through eight different electronic databases. Table 4.16 lists the search strings we used for each database.

Table 4.14: PICO and PICo definitions for the SLR about data integration in data lakes
Source: Couto and Ruiz (2022)

PICO	PICo
Population: Data lakes **Intervention**: Data integration **Comparison:** - **Outcome:** Models, metrics, evaluation, and challenges	**Population:** Data lakes **Interest:** Models, metrics, evaluation, and challenges **Context:** Data integration

Table 4.15: Control Paper for the SLR about data integration in data lakes
Source: Couto and Ruiz (2022)

Control Study 01	Zhu, Erkang, et al. "Josie: Overlap set similarity search for finding joinable tables in data lakes." International Conference on Management of Data (Zhu et al., 2019).

Table 4.16: Search strings for each electronic database for the SLR about data integration in data lakes
Source: Couto and Ruiz (2022)

Electronic Database	Search String
ACM	"query": AllField:(("data integration" AND "data lake∗")) "filter": ACM Content: DL
arXiv	"data integration" AND "data lake∗"
Google Scholar	allintitle: "data integration" "data lake∗"
IEEE Xplore	("All Metadata":"data integration") AND ("All Metadata":"data lake∗")
Science Direct	Title, abstract, keywords: "data integration" AND ("data lakes" OR "data lake")
Scopus	TITLE-ABS-KEY ("data integration" AND "data lake∗")
Springer	https://link.springer.com/search?dc.title=%22data+integration%22+AND+%22data+lake∗%22&date-facet-mode=between&showAll=true#
Web of Science	https://www.webofscience.com/wos/woscc/summary/703bae45-ee46-4661-ac06-b5aa14160e54-1b6bd85d/relevance/1 OR "data integration" AND "data lake∗" (All Fields)

Because this SRL returned fewer records if compared to the other two reviews, we did not use the Kappa method, since one reviewer selected and read all of the papers and the second reviewed all the process and selection.

SLR Results

In this section, we present our analysis for the papers, and we answer our five RQ. We retrieved in total 298 papers, and, after reading the title, abstract, and keywords, we got 82 papers to fully analyze. Finally, we accepted 22 papers following the SLR process. Table 4.17 shows the number of papers we retrieved and the ones we accepted per database. This Table shows that most of the papers came from ACM and Web of Science.

The accepted papers were published between the years 2018 to 2021, mostly from 2018 and on (Figure 4.3). We accepted ten journal papers and 12 conference papers.

Table 4.17: Papers per electronic databases for the SLR about data integration in data lakes

Source: Couto and Ruiz (2022)

Electronic Database	Initial	Accepted papers
ACM	68	5 papers: Hai and Quix (2019), Helal et al. (2021), Zhang and Ives (2019), Brackenbury et al. (2018), Zhu et al. (2019)
arXiv	2	0 paper
Google Scholar	2	1 paper: Dabbèchi et al. (2020)
IEEE Xplore	11	3 papers: Alserafi et al. (2016), Dong et al. (2021), Yang et al. (2020)
Science Direct	132	2 papers: Dhayne et al. (2021), Quinn et al. (2020)
Scopus	50	4 papers: Alili et al. (2017), Rezig et al. (2021), Hai et al. (2018), Koutras (2019)
Springer	16	2 papers: Jovanovic et al. (2021), Beyan et al. (2016)
Web of Science	17	5 papers: Alrehamy and Walker (2018), Pomp et al. (2018b), Endris et al. (2019), Haller and Lenz (2020), Kathiravelu and Sharma (2017)

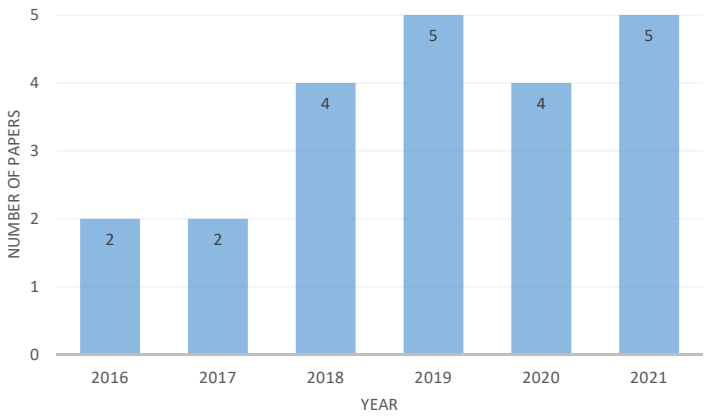


Figure 4.3: Papers per year - SLR about data integration in data lakes
Source: Couto and Ruiz (2022)

4.4.1 Models for data integration in data lakes

We decided to group the models into six groups: Graph-related, Query processing-based, Data profiling-based, Schema matching, Set similarity-based, and Layered architectures. We group the papers based on the title, abstract, keywords, and important terms in the document. Next, we present the groups and related models.

Graph-related

Koutras (2019) developed Data as a Language (Daal), which first transforms data into a graph, then create documents from the graph. From the documents they create embeddings, to find semantic relationships (for instance, how two columns from different tables could be joined). Haller and Lenz (2020) infer the schema of the data based on the SQL queries performed over data, using, for instance, the joins as points for data integration, to create a knowledge graph. Jovanovic et al. (2021) developed an integration manager that generates source-specific metadata when a new data source is registered in their system, which contains a Global Schema Building, for semi-automatic schema alignment and for the data merging process. They rank the candidates to match with terms in a global graph, based on a confidence level representing the degree of similarity between the concepts. The suggestion can be rejected or accepted by the user. The accepted matches are then defined as "sameAs" edges, compared to the similar concepts in the graph. Alrehamy and Walker (2018) developed an ontology-based data integration system named SemLinker. The system is responsible for extracting and maintaining the metadata, handling schema evolution, and finding mappings between the metadata and an ontology.

Query processing-based

Hai and Quix (2019) developed an approach that translates a subset of second-order tuple-generating dependencies (SO TGDS) into logically equivalent nested TGDS. The approach allows data integration through mapping dependencies. Endris et al. (2019) developed Ontario, an engine for federated query processing over heterogeneous data in data lakes. Hai et al. (2018) developed Scalable Query Rewriting Engine (SQRE), based on Apache Spark to translate queries for a logical representation, parse the queries according to the source to be queried, and execute queries in different data stores in a data lake, to present the integrated results. Alili et al. (2017) developed a model to enrich datasets by searching for related information in external data sources - a service lake. They use it, for instance, to add missing information to the dataset. Zhang and Ives (2019) developed an architecture based on Jupyter notebooks using the model they developed (JUNEAU) as backend and to extend the user interface. The backend integrates key-value stores and relational databases to capture and index data of interest. For data integration, the user can select a table and the system returns a ranked list of conceptually related tables that could be joined. Dhayne et al. (2021) developed EMR2vec, a platform to link clinical trial data and patient data, to help find the most suitable patients for clinical trials based on the eligibility criteria by querying an integrated data lake. Quinn et al. (2020) their solution is an integration technique to map the time-series data from a building Internet of

Things (IoT) sensor network to Facility Management-enabled Building Information Models (FM-BIMs), using Apache Cassandra as storage where the data can be queried.

Data profiling-based

Alserafi et al. (2016) developed a prototype for a metadata management system called Content Metadata for Data Lakes (CM4DL), which detects joinable data attributes between datasets. They also use data profiling techniques to describe each attribute and its type of data. The input for their algorithm is the files, a JSON containing metadata features created by the data profiling, and the threshold for matching datasets and attributes, and the output is the discovered relationships. Helal et al. (2021) developed a model named KGLac, that bases on a data profiling system on top of Apache Spark. KGLac uses embedding similarity search to reveal columns or tables that have a similar representation, enabling joining tables. They use data profiling to detect relations based on the content of data, such as inclusion dependency and primary and foreign key discovery.

Schema matching

Brackenbury et al. (2018) developed a similarity-based approach, to find related datasets in data swamps, based on schema matching and discovering techniques. Dabbèchi et al. (2020) use ELT jobs in "Talend open studio for big data" for schema mapping and integration of the different data sources: Facebook, Youtube, and Twitter. Data is stored in MongoDB and Cassandra NoSQL databases. Rezig et al. (2021) developed DICE (Data Discovery by Example) where the user provides examples of records in a data lake, and then the system suggests Primary Key/Foreign Key join paths, that can relate to other tables, based on exact or similarity matching. The candidates for PK/FK joins are then validated by the user. Dong et al. (2021) developed the PEXESO framework for joinable table discovery in data lakes. PEXESO uses pre-trained models to help transform textual attributes in high-dimensional vectors, so they join the tables using semantics.

Set similarity-based

Kathiravelu and Sharma (2017) propose Data Café, a data warehouse platform to create, integrate, and manage biomedical data lakes. They store the data in HDFS, MongoDB, and MySQL, the data schema is stored in Apache Hive, they use Apache Drill as SQL search engine, and they use Hazelcast as an in-memory data grid. For data integration, they identify join-attributes, which are the indexes that could lead from dataset A to dataset B, and then they create a graph-based o the intersection of the datasets. Yang et al. (2020) developed a model for the top-k set similarity joins (SSJOIN). They focus on the step size, which is the number of elements that are accessed in each algorithm’s

iteration. They developed a fixed size (l-ssjoin) and an adaptative size step algorithm (A-ssjoin). Zhu et al. (2019) developed JOSIE (JOining Search using Intersection Estimation), an overlap set similarity search algorithm that uses a search model to adapt according to the distribution of the data. They receive a column of a table as input, and they return the tables in the data lake that could be joined with the given columns, based on the largest number of distinct values.

Layered architectures

Beyan et al. (2016) proposed a data value chain based on 5 layers: Data Acquisition; Data Interpretation and Multilingual Interoperability; Data Analysis and Curation; Data Storage; and Data Usage. Pomp et al. (2018b) developed ESKAPE: a data ingestion, integration, and processing model formed by three layers: a Hadoop-based data lake, which contains the datasets, the semantic models, which are created during data ingestion for each dataset, and the knowledge graph, which combines all the semantic models into a unified repository similar to an ontology.

4.4.2 Similarity metrics for data integration

From the papers we selected, twelve papers do not mention the similarity metric they use to evaluate the similarity for data integration. Table 4.18 presents the similarity metrics most used in the papers.

Semantic similarity is the top-cited. Semantic similarity functions assign a score for the relationship between pieces of text by using a predefined metric (Alili et al., 2017; Pomp et al., 2018b; Helal et al., 2021; Dhayne et al., 2021).

The *Jaccard coefficient* is used in 3 papers. Jaccard is useful for comparing finite sets, represented by the quotient of the cardinalities of the intersection and the union of all tokens or characters in two strings (Jaccard, 1902).

MinHash-based approaches were used by Zhu et al. (2019), Brackenbury et al. (2018), (Alserafi et al., 2016). The latter states that it is an approach that makes comparisons of text n-grams, being a good approach for approximate string matching.

In its turn, the *Overlap set similarity*, present in two papers, represents the size of the intersection between two sets Zhu et al. (2019). The Cosine distance is also used in two papers and measures the similarity between two vectors by evaluating the cosine value of the angle between them (Alrehamy and Walker, 2018).

String-based measures, such as edit distance, are used by (Alrehamy and Walker, 2018), and Zhang and Ives (2019). The edit distance measures the dissimilarity between

two strings by counting the minimum number of operations that we need to perform to transform one string into the other.

Dong et al. (2021) used the *Euclidean distance*, which is the distance between two points, often used to check the similarity measure between time-series, and the *Jensen–Shannon divergence*, which measures the similarity between two probability distributions.

Alserafi et al. (2016) used *identity-based exact match*, where the attributes are normalized and then compared to find exact matches. It is a good approach for exact values comparison, such as numeric values.

Finally, Dhayne et al. (2021) use the *projection similarity*, that computes the level of similarity of a dataset in the dimensions of the features of the other dataset.

Table 4.18: Most used similarity metrics
Source: Couto and Ruiz (2022)

Similarity metric	N° of papers	Papers
Semantic similarity	4	Alili et al. (2017), Pomp et al. (2018b), Helal et al. (2021), Dhayne et al. (2021)
Jaccard coefficient	3	Brackenbury et al. (2018), Zhu et al. (2019), Yang et al. (2020)
MinHash-based distances	3	Zhu et al. (2019), Brackenbury et al. (2018), (Alserafi et al., 2016)
Overlap set similarity	2	Zhu et al. (2019), Yang et al. (2020)
Cosine distance	2	Yang et al. (2020), Alrehamy and Walker (2018)
String-based measures	2	(Alrehamy and Walker, 2018), Zhang and Ives (2019)
Euclidean distance	1	Dong et al. (2021)
Jensen–Shannon divergence	1	Dong et al. (2021)
Identity-based exact match	1	Alserafi et al. (2016)
Projection similarity	1	Dhayne et al. (2021)

4.4.3 Evaluation of data integration models for data lakes

From the 22 papers, eight do not present an evaluation of their approaches.The papers that evaluate their studies perform the evaluations based on the following metrics:

- Scalability - 7 papers: Kathiravelu and Sharma (2017); Hai et al. (2018); Brackenbury et al. (2018); Yang et al. (2020); Quinn et al. (2020); Dong et al. (2021); Alrehamy and Walker (2018). It represents the ability to deal with a crescent amount of data.

- Execution time - 6 papers: (Endris et al., 2019; Alrehamy and Walker, 2018; Yang et al., 2020; Alserafi et al., 2016; Hai et al., 2018; Zhu et al., 2019). Those papers present the average time to run their experiments.
- Precision - 6 papers: Jovanovic et al. (2021); Alrehamy and Walker (2018); Alserafi et al. (2016); Helal et al. (2021); Dong et al. (2021); Dhayne et al. (2021). Precision measures the correct answers of the model over the total number of observations.
- Recall - 4 papers: Helal et al. (2021); Dong et al. (2021); Alserafi et al. (2016); Jovanovic et al. (2021). It measures the true positives over the sum of the true positives plus the false negatives.
- F1 - 2 papers: Alserafi et al. (2016); Helal et al. (2021). F1 score is the harmonic mean between precision and recall.
- Accuracy - 2 papers: Brackenbury et al. (2018); Alrehamy and Walker (2018). It measures the proportion of correctly predicted observations regarding the total number of observations.

Other types of evaluation were also cited. For instance, Haller and Lenz (2020) evaluated their model based on the ability to reconstruct the data schema based on the SQL queries. Jovanovic et al. (2021) measured the usability and the number of times the user had to interneve to find the matches manually. Hai and Quix (2019) evaluated the correctness, completeness, and performance of their model, while Endris et al. (2019) evaluated the cardinality (number of answers a query returns), completeness (query results percentage compared to another engine), and *dief@t* (measures the continuous engine's efficiency). Yang et al. (2020) measured the number of candidates for joining according to the threshold. Zhu et al. (2019) measured the number of top results to retrieve based on other solutions; and Hai et al. (2018) evaluated their model's functionality.

4.4.4 Type of data they integrate

The types of data most used in the experiments to validate the models are CSV-like and relational tables. *CSV, TSV, or other tabular formats* are presented in 8 papers (Alrehamy and Walker, 2018; Jovanovic et al., 2021; Quinn et al., 2020; Zhang and Ives, 2019; Endris et al., 2019; Zhang and Ives, 2019; Koutras, 2019; Hai and Quix, 2019; Hai et al., 2018). Seven papers also present the use of *relational tables* (Alili et al., 2017; Zhang and Ives, 2019; Haller and Lenz, 2020; Rezig et al., 2021; Dong et al., 2021; Dhayne et al., 2021; Kathiravelu and Sharma, 2017) The *JSON format* is used by 4 papers (Alrehamy and Walker, 2018; Jovanovic et al., 2021; Pomp et al., 2018b; Endris et al., 2019).

Three papers present the use of *social media data* (Alrehamy and Walker, 2018; Yang et al., 2020; Beyan et al., 2016). Other two papers discuss the use of entire *data lakes*, such as OpenData, OpenML datasets, and WebTables Zhu et al. (2019); Alserafi et al. (2016). Three papers base their experiments on *XML files* (Endris et al., 2019; Hai et al., 2018; Dhayne et al., 2021), and two others use HTML files (Brackenbury et al., 2018; Dong et al., 2021).

Other types of data format include: *collections of documents* (Dabbèchi et al., 2020), *NoSQL databases* such as MongoDB (Hai et al., 2018), a *file system dump* (Brackenbury et al., 2018), RDF (Endris et al., 2019) and other *domain-specific datasets* (Beyan et al., 2016; Helal et al., 2021; Yang et al., 2020).

Surprisingly, we did not find papers that report the use of common types of data in data lakes, such as Parquet, Avro, and ORC.

4.4.5 Challenges in data integration in data lakes

From the selected papers, nine papers present some challenges related to data integration in data lakes. Figure 4.4 presents a word cloud for the challenges.

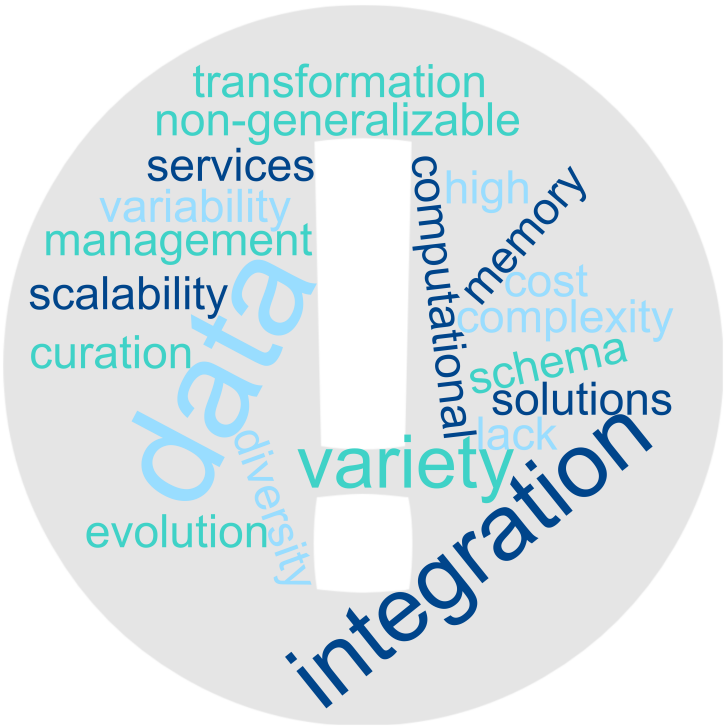


Figure 4.4: Word Cloud for the challenges of Data integration in Data lakes
Source: Couto and Ruiz (2022)

- Complexity (Koutras, 2019): They discuss the challenge of transforming data, including challenging in construction, incorporating information about the schema, and

capturing entries from semi-structured datasets, which can contain comments or messages generated by a system.

- Computational cost (Dong et al., 2021): To compute similarity for high-dimensional data is expensive. Checking whether the tables are joinable or not is also time-consuming because of the large number of tables in a data lake.
- Diversity (Hai et al., 2018): There is a high diversity in the data management in data lakes, with many different solutions and frameworks, but they are not always easily integrated among them.
- In-memory integration (Hai et al., 2018): answering a single query based on data from several sources without creating a new structure to join the data.
- Lack of available solutions (Beyan et al., 2016): the author states that there is a lack of data integration and curation services for big data.
- Non-generalizable solutions (Kathiravelu and Sharma, 2017): The solutions are more specific for a certain data source our format than generalizable, and the solutions usually expect that the users know the data schema or storage format, and it is a problem, for instance, for medical data since they usually are consisted by a huge number of small datasets.
- Scalability (Alrehamy and Walker, 2018): the ability of a system to be prepared to efficiently handle more data.
- Variability (Alserafi et al., 2016; Alrehamy and Walker, 2018): it represents changes that occur in data schema and structure; the schema evolution in big data.
- Variety (Dhayne et al., 2021; Alserafi et al., 2016; Dabbèchi et al., 2020; Alrehamy and Walker, 2018; Dhayne et al., 2021): the data are difficult to analyze since it is mostly unstructured or semi-structured. Because of the variety, we find syntactic and semantic complexity of the different datasets, for example, differences in the semantic concepts, which can be more generic or more specific.

4.4.6 Conclusions of the SLR about data integration in data lakes

This section presented a systematic literature review to retrieve the state-of-the-art related to data integration on data lakes. Our initial set of paper is composed of 298 papers, and, after selection, we ended up having 22 papers accepted, published between 2018 and 2021. We identified six groups of related papers according to the models: Graph-related, Query processing-based, Data profiling-based, Schema matching, Set similarity-based, and Layered architectures. We also identified the most used similarity metrics for

data integration, such as semantic similarity, Jaccard, MinHash-based, Overlap, Cosine, and String-based measures. Additionally, we investigated how they evaluate their models, and most of them perform experiments to check the scalability, execution time, and precision. Among the types of data they integrate, we found that CSV-like and relational tables are the most popular. Finally, we mapped nine challenges related to data integration in data lakes: complexity, computational cost, diversity, in-memory integration, lack of available solutions, non-generalizable solutions, scalability, variability, and the most cited: variety.

Remarks

In this section, we presented three literature reviews containing an overview of the work related to ours: a mapping study about data lakes and two systematic literature reviews, the first one about data profiling and the second one about data integration in data lakes. The MS identifies the most common definitions for the term data lake, which leads us to generate a new formalization for the term. We also trace the possible architectures for data lakes, where we identify that the Hadoop ecosystem is the basis for the most used architectures. In the SLR about data profiling, we generate an overview of data profiling in big data, focusing on the challenges related to the field. Finally, we perform an SLR about data integration in data lakes, where we identify the trending topics, and the most recent works closely related to our model, and the open challenges. Some papers were retrieved in more than one literature review. For instance, Hai et al. (2018); Koutras (2019) were retrieved in the first MS and the last SLR, while Alserafi et al. (2016) was retrieved in both SLR. It suggests that we searched in a very specialized group of researchers contributing to the data lake scientific community.

Both SLR presented challenges that are partially solved in the papers we identified. In Chapter 5, Section 5.6, after presenting the model we developed, we will recap the challenges we here identified, comparing what is achieved by the papers retrieved from the SLR with the results from our model.

5. A MODEL FOR AUTOMATIZED DATA INTEGRATION IN A HADOOP-BASED DATA LAKE

This chapter presents the model we created for automating data integration and our experiments with bioinformatics datasets. We started by explaining the preliminaries, with the system architecture for the data lake and the datasets to be used in the experiments. Next, we present our model composed of data ingestion & storage, processing & integration, and presentation processes. The next step includes a discussion about our results, a presentation of a use case, the complexity, and how we evaluate our model. We end the chapter by comparing our work to the state-of-the-art and discussing final remarks.

5.1 Preliminaries

This section presents the architecture we built to support our model, and the characteristics of the datasets we used in our experiments.

5.1.1 System deployment architecture

Our data lake is supported by an Ubuntu 20 64-bit Linux server, having the following configuration: 16GB RAM DDR3, Processor Intel® Core(R) I7-4790 CPU@3.6GHz x 8, 1TB disk capacity. The data lake is composed of ten components:

1. Apache Nifi: the framework we used for data ingestion.
2. Python - Jupyter Notebook: a programming language and a web application to run Python code, used for data processing.
3. Neo4J: a graph database used to visualize the integration among the dataframes.
4. Hadoop Namenode: the master node in the HDFS architecture.
5. Hadoop History Server: keeps the logs of all the jobs that ran on Hadoop.
6. Hadoop Resource Manager: contains the YARN (Yet Another Resource Negotiator), a service that manages the resources and schedules/monitors the jobs.
7. Hadoop Node Manager: launches and manages the containers on a node.
8. Hadoop Datanodes: Three containers (Datanode 1, Datanode 2, and Datanode 3). The workers nodes in the HDFS architecture, where the data is actually stored.

Appendix A presents details about how to configure the Hadoop-based data lake (adapted from Europe (2021)), to ease replicability.

5.1.2 Datasets

Our study selected eight bioinformatics datasets to populate our data lake and work on automatized data integration, described below. The datasets were indicated by a specialist in bioinformatics and computer science, Professor Anil Wipat[1]. He is a Professor of Bioinformatics in the School of Computing Science at Newcastle University, where part of this research took place.

- OMIM (McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins University (Baltimore, MD), 2021): Online Mendelian Inheritance in Man - human genes and genetic phenotypes. We used the *genemap2* dataset.
- DISGENET (Piñero et al., 2020): Collections of genes and variants associated with human diseases.
- REACTOME (Jassal et al., 2020): We are using the *UniProt2Reactome* dataset. It is composed of reactions, proteins, and pathways.
- MONDO (Mungall et al., 2017): Ontology for disease definitions.
- DRUGBANK (Wishart et al., 2006): Pharmaceutical knowledge base, we split it into two dataframes: DRUGBANK and DRUGBANK_PROTEIN.
- IID (Kotlyar et al., 2019): Integrated Interactions Database - database of detected and predicted protein-protein interactions. We used the human data annotated dataframe.
- DRUGCENTRAL (Avram et al., 2021a): Online drug compendium - we use the drug-target interaction dataset.
- UNIPROT (Consortium, 2019): We are using the *reviewed Swiss-Prot XML* dataset, a non-redundant and manually annotated database containing protein sequences.

Table 5.1 presents the characteristics of each dataset, ordered by size from the smaller (DRUGBANK) to the larger (IID). Table 5.1 shows that we selected heterogeneous datasets, having varied sizes (from 1 MB to 1,8GB), from 13k entries to almost 1 million entries, with the number of attributes varying from 6 to 253. The datasets are also presented in different formats, such as TXT, XML, TSV, JSON, and via API.

[1]More information at: http://homepages.cs.ncl.ac.uk/anil.wipat/. Accessed December, 2021.

Table 5.1: Characteristics of the bioinformatics datasets
Source: The author (2022)

Dataset	Size (MB)	Entries	Attributes	Format
DRUGBANK	0,95	13580	9	XML
DRUGBANK PROTEIN	1,40	26965	7	XML
OMIM	1,80	17092	14	TXT
DRUGCENTRAL	2,50	17390	19	TSV
MONDO	4,00	43233	13	JSON
DISGENET	10,30	84037	16	TSV
UNIPROT	30,20	565255	7	API
REACTOME	37,90	826877	6	TXT
IID	1800,00	975877	253	TXT

We can see we have heterogeneous datasets since we have different formats (XML, TXT, TSV, JSON, and query by API), varied sizes (from 1 MB to 1,8GB), a different number of entries (from 13k to 975k), and a diverse number of attributes (from 6 to 253) for each dataset.

5.2 Data Lake Processes in the Data Integration Model

Our model is composed of data ingestion, storage, processing & integration, and presentation processes (see Figure 5.1). Next, we detail each process.

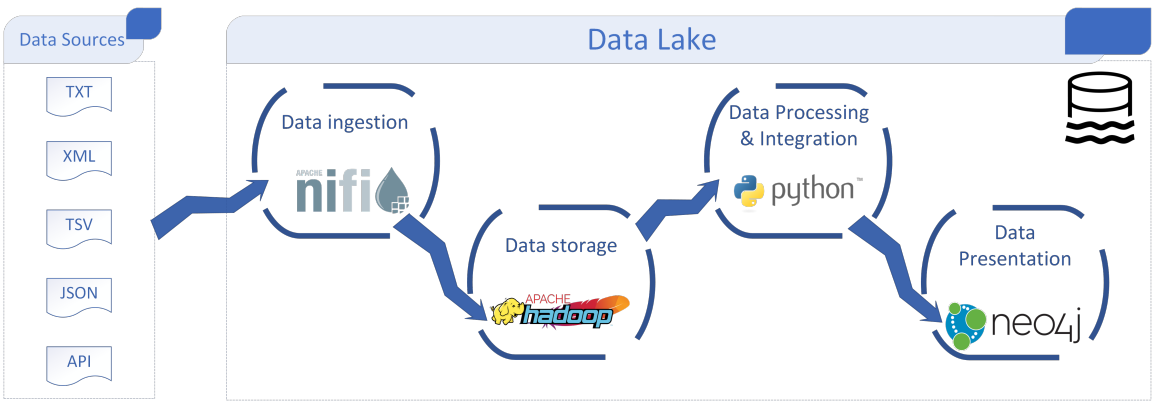


Figure 5.1: Composition of the data lake
Source: Couto et al. (2022a)

5.2.1 Data Ingestion & Storage

To be able to integrate the dataframes, we first need to ingest and store the data inside the data lake. Since data ingestion and storage are closely related, we present both processes together. We ingested data into the data lake by creating processors in Apache Nifi. We create one group of processes for each dataset, where the process searches for the dataset on HTTP (for MONDO, REACTOME, DISGENET, IID, and DRUGCENTRAL), or in a local folder for OMIM and DRUGBANK because they are not available directly due to the need of registering and licensing. Then we unzipped some of the datasets and renamed them all for standardization. Next, Apache Nifi sends the datasets to Hadoop to be stored in HDFS. For UNIPROT, we use the API directly on Jupyter notebooks.

Figure 5.2 presents the home page when accessing Apache Nifi, having the seven processes groups responsible for getting our datasets into Hadoop. We collect the datasets using different Nifi processors, detailed above:

- ListFile: Fetches the files from the local filesystem. For this processor, we configure the properties:
  - Input Directory: /datasets
  - Input Directory Location: Local
  - File Filter: the name of the dataset (OMIM_genemap2.txt)
- InvokeHTTP: Connects with a HTTP Endpoint. For this processor, we configure the properties:
  - HTTP Method: GET
  - Remote URL: URL to the file
- FetchFile: Reads the contents of a file and streams it.
- UnpackContent: Unpacks the FlowFile contents.
  - Packaging Format: zip (also accepts tar and flowfiles)
- CompressContent: Decompresses or compresses FlowFile contents based on user-specified algorithms. The properties we configure are:
  - Mode: decompress
  - Compression Format: gzip (also accepts bzip2, xz-lzma2, lzma, snappy, etc.)
- UpdateAttribute: Updates or deletes attributes based on a regular expression. We configure the property:
  - filename: name of the file to be stored in Hadoop.

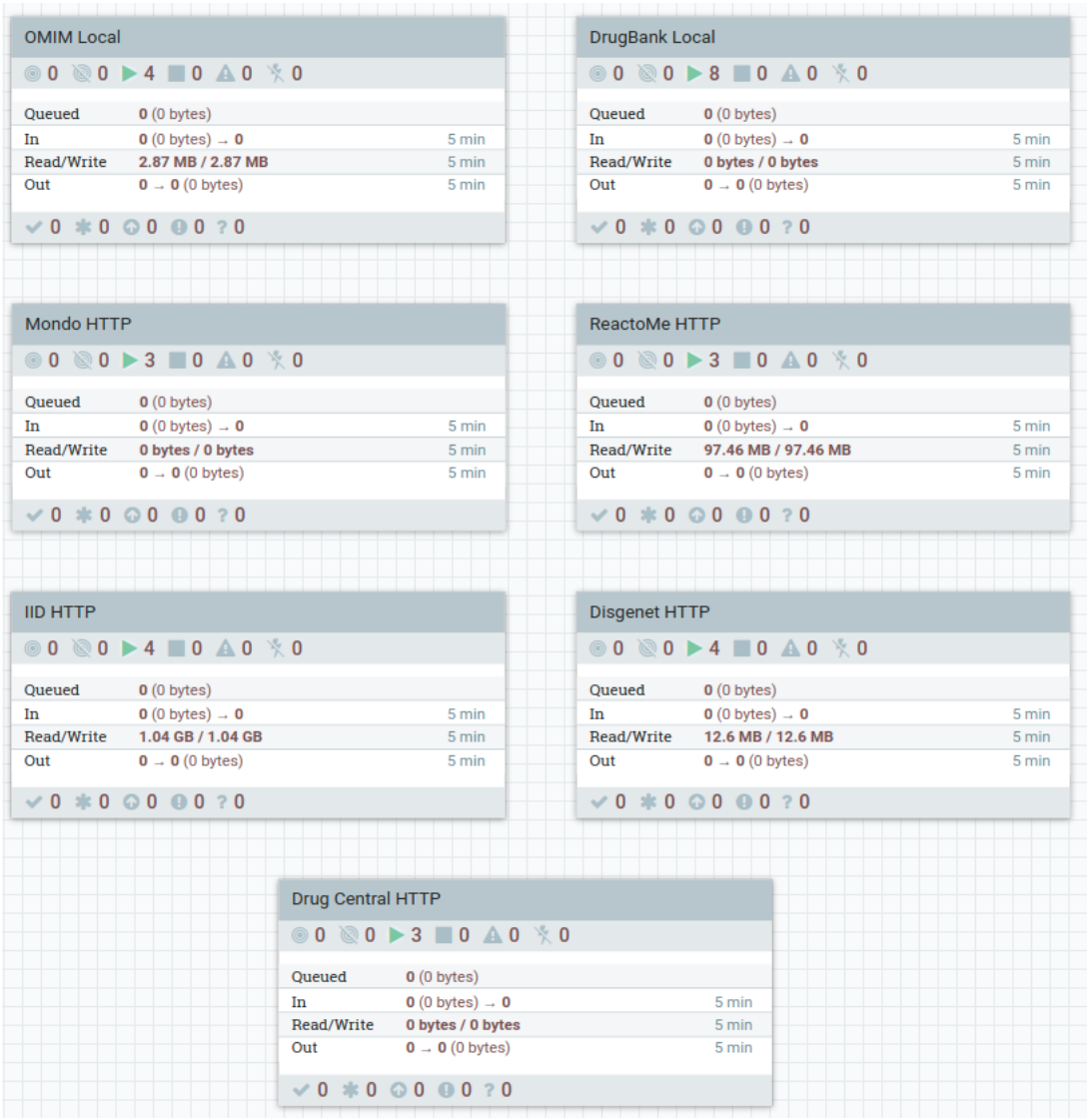


Figure 5.2: Groups of processors created on Apache Nifi for data ingestion
Source: The author (2022)

- PutHDFS: Writes the FlowFile data to HDFS. In this processor we set the following attributes:

  – Hadoop Configuration Resources: /etc/conf/core-site.xml

  – Directory: /datasets

  – Conflict Resolution Strategy: replace. Other options include *ignore*, *fail*, or *append*.

Figure 5.3 presents an example of a process group for the OMIM dataframe. We can also specify the Nifi Schedule to get the file for all the processors. For instance, we can use scheduling parameters to get the files once per day, once per week, or each *n* hours. Below we present the list of processors we used for each dataset.

- OMIM: ListFile → FetchFile → UpdateAttribute → PutHDFS

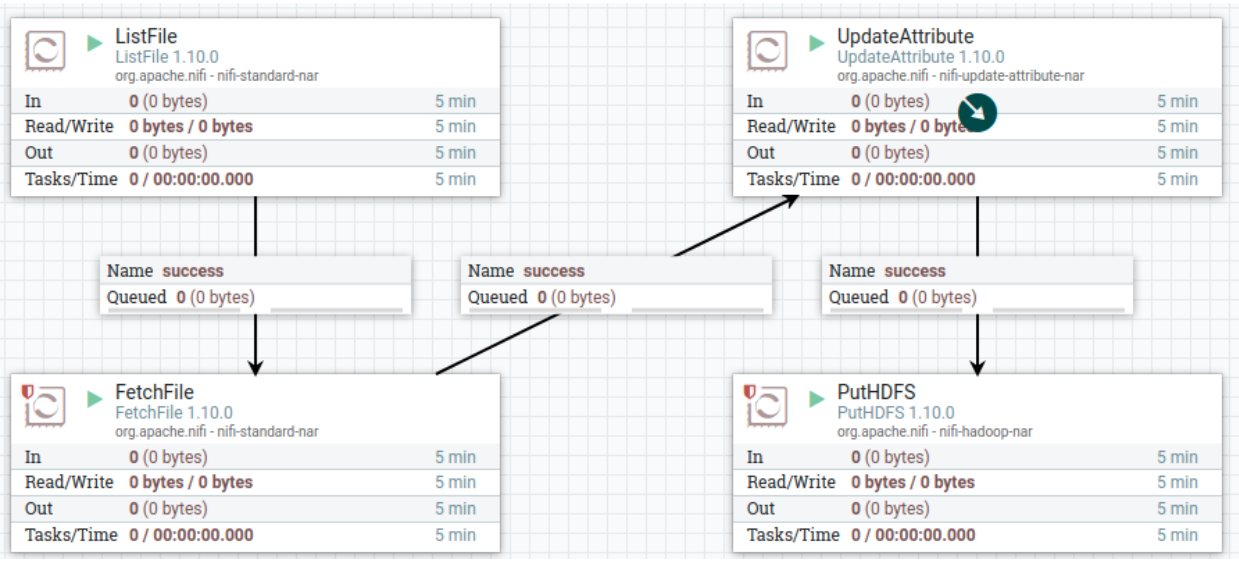


Figure 5.3: Apache Nifi - OMIM group of processors
Source: The author (2022)

- DRUGBANK: ListFile → FetchFile → UnpackContent → UpdateAttribute → PutHDFS
- MONDO, REACTOME, DRUGCENTRAL, and DRUGBANK (json): InvokeHTTP → Update-Attribute → PutHDFS
- IID and DISGENET: InvokeHTTP → CompressContent → UpdateAttribute → PutHDFS

We store the data in Hadoop HDFS. We configure Hadoop to have three datanodes for data replication. It means that, in a Hadoop traditional architecture, each data block is replicated on each of the datanodes, which helps speed up the data processing.

5.2.2 Data Processing & Integration

We start data processing using the Python - Jupyter Notebook. We start the experiments by turning the datasets into Python Pandas dataframes. Pandas is a Python library for data analysis and manipulation. The standardization of the datasets as dataframes assure a unified entry for the algorithm, solving issues regarding one dataset being derived under one condition and the others being on other conditions. We also use other libraries, such as *HDFS*, that provide a pure HDFS client, *bioservices* that provide API access to UNIPROT, and the package *py_stringmatching* that implements the similarity metrics.

After creating the dataframes, a specialist in data science analyzed the datasets to manually map the attributes candidates for points of integration. To create the DRUGBANK dataframe, we based on the solution provided by Himmelstein (2016). To perform the manual mapping of the integration points, the specialist analyzed the names of the

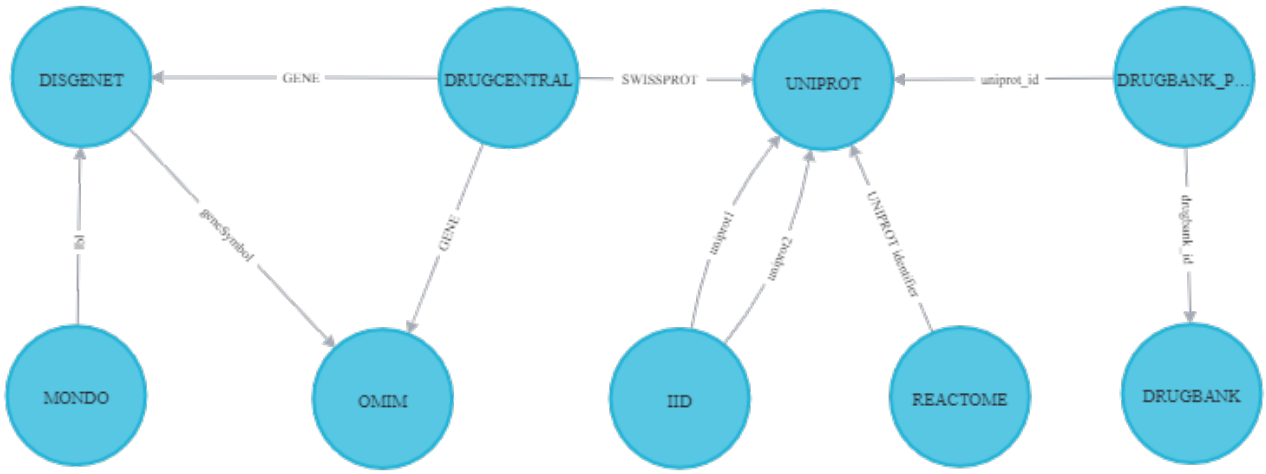


Figure 5.4: Manually mapped integration
Source: Couto et al. (2022a)

columns and a sample of data for each column, using data profiling techniques. The specialist took about four hours to finish the analysis, and we present the manual mapping in Figure 5.4. Figure 5.4 presents the manual data integration points, based on a graph visualization, where the nodes or vertices are the names of the dataframes, and the edges are the attributes' names. The orientation of the arrow indicates that, for instance, the attribute 'lbl' from the MONDO dataframe is a point of integration to the dataframe DISGENET, meaning that a percentage of 'lbl' is also present in another attribute of DISGENET. We developed this Figure to be later compared with the results of the algorithm we developed for data integration so that we could compare a user specialist analysis with the algorithm output.

Data Integration Algorithm

Our data integration algorithm is based on the concept of intersection or overlap between two attributes in sets of data. We first identify the unique values of each attribute for each dataframe. Then we compare each dataframe attribute with all the other dataframe's attributes to check if the unique values of the content of each attribute are contained in any other attributes of all of the other dataframes. The attribute with fewer unique values indicates the orientation of the data integration. For instance, let us analyze the following case that includes dataframes (df) and attributes (att):

- df1['att01'] has 10 unique values;
- df2['att06'] has 20 unique values;
- 10 values from df1['att01'] are also present on df2['att06'].

In that case, we can notice that 100% of df1['att01'] are also present in df2['att06'], being that a good point for data integration. The notation would be: df1['att01'] $\longrightarrow$

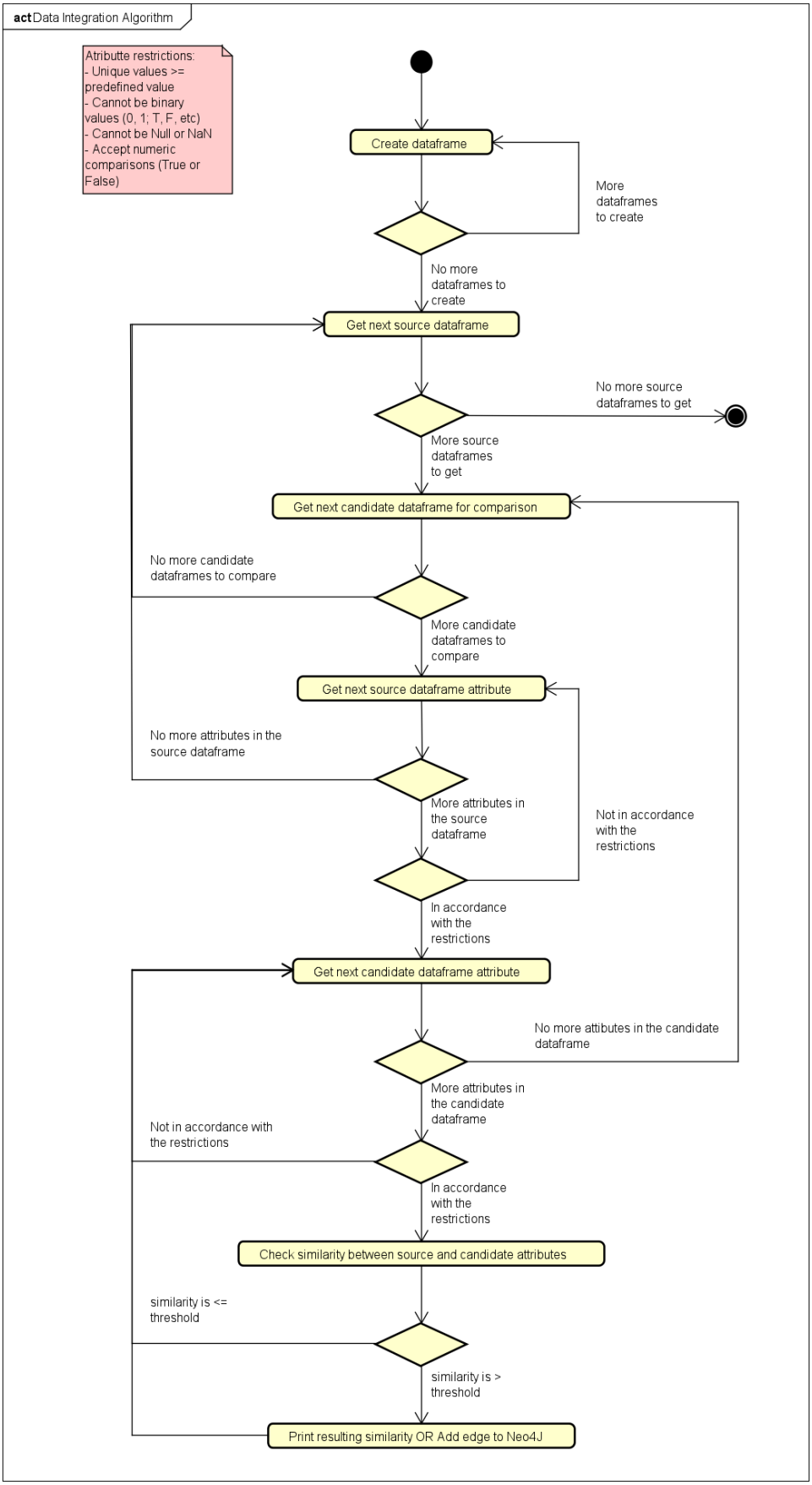


Figure 5.5: Algorithm for data integration in UML Activity Notation
Source: Couto et al. (2022a)

df2[’att06’]. Regarding the minimum value for data intersection, we defined a threshold of 0.4 (in a range from 0–1)] to identify good integration points, but it is configurable according to the user’s needs. It means that if two columns in a dataframe have 40% or more of data in common, the two columns are candidates for data integration, and the dataframes where the columns come from are integrable.

To define the best threshold for our experiments, we tested different values and compared the results with the specialist’s analysis. We started with 0.9, and after each execution, we compared our results with the specialist’s manual integration. When we noticed that the selected threshold retrieved at least all of the integration points defined by the specialist, we stopped decreasing the threshold, determining the value of 0.4.

Figure 5.5 details the activities diagram for the algorithm we developed. Figure 5.5 shows that we can configure restrictions to select the attributes to be analyzed, such as the minimum number of unique values that an attribute must have to enter in the comparisons, and if we want to perform comparisons with attributes that contain only numeric values. Other restrictions include: removing attributes with only nulls or NaN and removing attributes with binary values (0 or 1, T or F). The binary values would not present real candidate points for data integration among the datasets since they mostly represent True or False values. For instance, the IID dataset presents dozens of attributes that are named after diseases, where the value = 0 corresponds to False and values = 1 corresponds to True (e.g.: ’bone disease’, ’overnutrition’, ’asthma’, ’lymphoma’).

The algorithm starts by creating dataframes for all the datasets, then it selects the first source dataframe, which will be compared to the other dataframes. Then the attributes of the source dataframe are compared with the attributes of the first candidate dataframe to check the similarity. It happens until we do not have more source dataframes to be compared to the candidates.

Our algorithm also handles so that there are no redundant comparisons among dataframes and attributes. Firstly, we assure that a dataframe is not compared to itself by identifying its previously defined name in the algorithm. Secondly, when we compared each attribute of the first dataframe, we stored its description in a variable. Before comparing the dataframe’s attribute with another, we check that there are no attributes with the same description. Therefore, we exclude the possibility of redundant comparisons between dataframes and attributes.

We based our work on an in-memory set similarity approach since the integration is executed using Python notebooks. As for similarity metrics, we use the most well-known distance measures for sets according to Ontañón (2020): Tverski’s (Tversky, 1977), Sørensen Dice’s index (Sørensen, 1948), and Jaccard (Jaccard, 1902), compared to the Szymkiewicz-Simpson Overlap Coefficient (Vijaymeena and Kavitha, 2016).

Our algorithm returns a list having the names of the dataframes, attributes, and resulting values for the Szymkiewicz-Simpson overlap coefficient – Equation 5.1, which is

the main result, compared to other similarity metrics (Jaccard – Equation 5.2, Sørensen-Dice – Equation 5.3, and Tversky – Equation 5.4). The resulting values for the similarity metrics range from 0 (attributes are not at all similar) to 1 (attributes contain the same data). Next, we present the equations related to the metrics, where X represents the attribute of the source dataframe and Y represents the attribute of the dataframe candidate to be compared, according to the *py_stringmatching* library documentation (py_stringmatching Team, 2021).

The Overlap Equation calculates the size of the intersection divided by the smaller of the size of the two attributes or sets:

$$overlap(X, Y) = \frac{|X \bigcap Y|}{min(|X|, |Y|)} \tag{5.1}$$

The Jaccard measures the size of the intersection divided by the size of the union:

$$jaccard(X, Y) = \frac{|X \bigcap Y|}{|X \cup Y|} \tag{5.2}$$

The Sørensen-Dice similarity score returns twice the intersection divided by the sum of the cardinalities.

$$dice(X, Y) = \frac{2 \times |X \bigcap Y|}{|X| + |Y|} \tag{5.3}$$

The Tversky index is a generalization of the Sørensen-Dice's and the Tanimoto coefficient (aka Jaccard index) coefficient, but introduces the use of the parameters α and β, where $\alpha = \beta = 1$ produces the Tanimoto coefficient and $\alpha = \beta = 0.5$ produces the Sørensen–Dice coefficient:

$$tversky(X, Y) = \frac{|X \bigcap Y|}{|X \bigcap Y| + \alpha|X - Y| + \beta|Y - X|}; \alpha, \beta >= 0 \tag{5.4}$$

Now let us present how the data integration algorithm is incorporated in the data lake processes.

5.2.3 Data Presentation

We create a database on Neo4J, where the nodes are the dataframes, and the edges are the name of the attributes. It helps us to visualize the relationships between the dataframes better.

After analyzing the first results presented by the algorithm, we identified that some suggested integration points are numeric values that, in our dataframes, do not

represent actual data integration points - thus were considered as false positives. For instance:

- UNIPROT['Lenght'] it is the length of the canonical sequence and it varies from 3 to 4 numeric chars;
- OMIM['Entrez_Gene_ID'] the National Center for Biotechnology Information (NCBI) gene ID, values from 1 to 115029024;
- DRUGCENTRAL['STRUCT_ID'] the structure ID, and has values from 1 to 5390;
- DISGENET['YearInitial'] and DISGENET['YearFinal'] are years from 1924 to 2020;
- DISGENET['NofPmids'] the PubMed id, and has values from 1 to 67;
- DISGENET['NofSnps'] the Single nucleotide polymorphisms (SNP) id, has values from 1 to 284.

That is why we decided to add a parameter in the algorithm do set if we want to make numeric comparisons. We set the parameter to false since, in our case, it does not represent actual data integration points, but to be able to generalize for different domains and different types of datasets, that kind of comparison must be useful.

Regarding the similarity metrics, as Tversky, Sørensen-Dice, and Jaccard present correlated values for our data (Tverski's index with α and β = 0.5 was equal Sørensen-Dice and twice the Jaccard coefficient), we show only the Jaccard and the overlap values in Table 5.2.

Figure 5.6 presents the final data integration resulting from our algorithm for the bioinformatics dataframes. In this graph visualization, similar to the manually mapped data integration visualization, the names of the dataframes are the vertices. The names of the attribute responsible for the integration are presented in the edges that connect the vertices. Figure 5.6 presents 9 nodes and 32 edges, some with only one edge between them (such as MONDO and DISGENET) and others having a high concentration of edges, such as IID, UNIPROT, and DRUGCENTRAL. The higher concentration of edges pointing to a dataframe means that the dataframe is referenced by a high number of other dataframes, meaning they represent an important point of integration.

5.3 Discussion

After carefully comparing the Jaccard and Overlap results with the manual mapping and reviewing the actual dataframes, we identified that the Overlap provides better insights about the relationships that could be created among the dataframes.

Table 5.2: Final data integration mapping
Source: Couto et al. (2022a)

DF1	Column DF1		DF2	Column DF2	Overlap	Jaccard	Manually mapped?
UNIPROT	Entry	←	REACTOME	UNIPROT identifier	0,409	0,058	✓
UNIPROT	Gene names	←	OMIM	Approved_Symbol	0,466	0,015	×
UNIPROT	Gene names	←	DISGENET	geneSymbol	0,483	0,009	×
UNIPROT	Gene names	←	IID	symbol2	0,484	0,017	×
UNIPROT	Gene names	←	DRUGCENTRAL	GENE	0,486	0,002	×
UNIPROT	Gene names	←	IID	symbol1	0,488	0,017	×
DRUGCENTRAL	ACCESSION	→	DRUGBANK_PROTEIN	uniprot_id	0,488	0,018	×
UNIPROT	Organism	←	DRUGBANK_PROTEIN	organism	0,499	0,206	×
DRUGCENTRAL	ACCESSION	→	IID	uniprot1	0,509	0,072	×
DRUGCENTRAL	ACCESSION	→	IID	uniprot2	0,510	0,071	×
DISGENET	geneSymbol	←	DRUGCENTRAL	GENE	0,528	0,110	✓
REACTOME	UNIPROT identifier	←	DRUGBANK_PROTEIN	uniprot_id	0,546	0,030	×
REACTOME	UNIPROT identifier	←	IID	uniprot2	0,565	0,107	×
REACTOME	UNIPROT identifier	←	IID	uniprot1	0,567	0,105	×
DRUGBANK_PROTEIN	uniprot_id	→	IID	uniprot1	0,583	0,147	×
DRUGBANK_PROTEIN	uniprot_id	→	IID	uniprot2	0,590	0,147	×
OMIM	Approved_Symbol	←	DRUGCENTRAL	GENE	0,609	0,082	✓
DRUGCENTRAL	GENE	→	IID	symbol1	0,615	0,076	×
DRUGCENTRAL	GENE	→	IID	symbol2	0,617	0,075	×
REACTOME	UNIPROT identifier	←	DRUGCENTRAL	ACCESSION	0,624	0,019	×
DISGENET	diseaseType	→	MONDO	lbl	0,667	0,000	✓
REACTOME	Species	→	DRUGCENTRAL	ORGANISM	0,750	0,051	×
UNIPROT	Entry	←	DRUGCENTRAL	ACCESSION	0,829	0,004	×
OMIM	Approved_Symbol	→	IID	symbol1	0,866	0,704	×
DISGENET	geneSymbol	→	IID	symbol1	0,880	0,464	×
OMIM	Approved_Symbol	→	IID	symbol2	0,885	0,717	×
DISGENET	geneSymbol	→	IID	symbol2	0,898	0,469	×
UNIPROT	Entry	←	DRUGBANK_PROTEIN	uniprot_id	0,909	0,008	✓
DISGENET	geneSymbol	→	OMIM	Approved_Symbol	0,915	0,536	✓
UNIPROT	Entry	←	IID	uniprot1	0,953	0,030	✓
UNIPROT	Entry	←	IID	uniprot2	0,960	0,031	✓
DRUGBANK	drugbank_id	←	DRUGBANK_PROTEIN	drugbank_id	1,000	0,579	✓

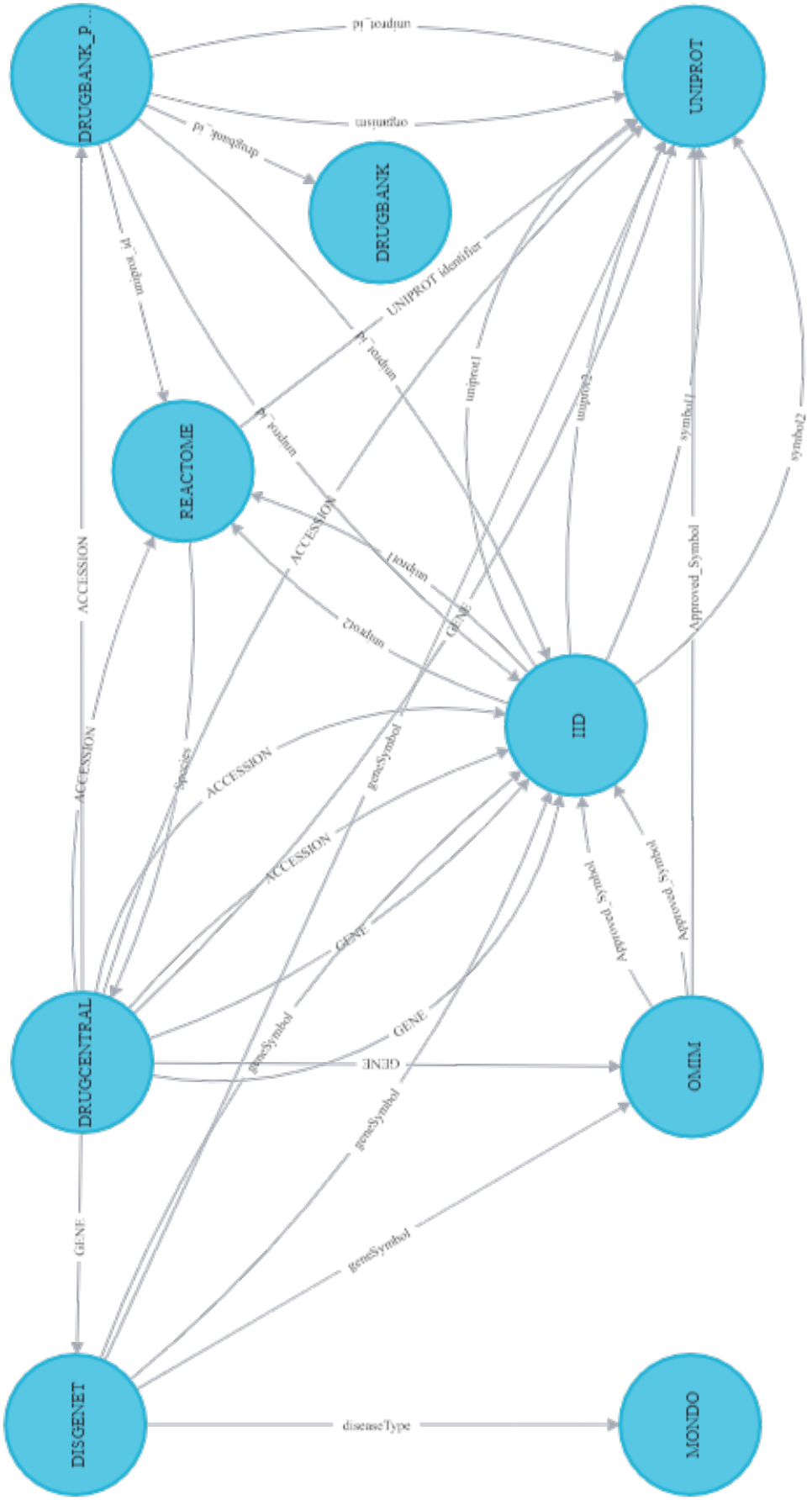


Figure 5.6: Final data integration
Source: Couto et al. (2022a)

For instance, for the relationship DISGENET["diseaseType"] and MONDO["lbl"], the Jaccard index is equal to zero, while the Overlap is 0,667. We checked the data, and we really found a point for integration in that case. Another example is DRUGBANK["drugbank_id"] and DRUGBANK_PROTEIN["drugbank_id"], which represent the same data according to the Overlap coefficient (1,000) and to our manual analysis, and in this case, the Jaccard index is 0,579.

Hence, the Overlap coefficient seems to represent better how similar two attributes are and the level of data integration we could achieve if we integrate two dataframes using the top-ranked pairs of attributes indicated by the algorithm. Thus, we decided that for our dataframes, it is better to use the Overlap Coefficient.

When we manually mapped the points for integration, we identified 10 points (see Figure 5.4), while our model identifies 32 points, presented in Table 5.2. Table 5.2 presents all the mannually mapped points of integration, but one: DRUGCENTRAL["SWISSPROT"] and UNIPROT ["ENTRY"]. For that integration point, our model shows that the coefficient for that integration is less than 0,4, while suggesting two better points: DRUGCENTRAL["GENE"] $\rightarrow$ UNIPROT["Gene names"], with a overlap of 0,487, and DRUGCENTRAL["ACCESSION"] $\rightarrow$ UNIPROT["ENTRY"], with an overlap of 0,829.

Additionally, our model discovered 22 more paths of integration that were not manually identified, which we list above:

- From IID and: DISGENET, DRUGCENTRAL, OMIM, REACTOME, and DRUGBANK_ PROTEIN
- From UNIPROT and: OMIM and DISGENET
- From REACTOME and: DRUGBANK_PROTEIN and DRUGCENTRAL
- From DRUGBANK_PROTEIN and DRUGCENTRAL

5.4 Use case

Let us discuss the utility of our model by considering the data integration example in the field of bioinformatics. A data scientist has access to a data lake with the same bioinformatics datasets we worked on: OMIM, DISGENET, REACTOME, MONDO, DRUGBANK, IID, DRUGCENTRAL, and UNIPROT. The data scientist received the task to study neglected diseases, such as tuberculosis. To do so, it is necessary to explore the data related to the gene *inhA*, which is related to the organism *Mycobacterium tuberculosis*. Having those two pieces of information, it is easy to find the related data on UNIPROT. Actually, it may be on the top-5 results of a quick search on Google. But how will the person know if and how the data found in UNIPROT can be integrated with the other data sources so that they

can find additional information? Well, usually, the person would have to put an effort into understanding the schema of all the datasets, analyze the data dictionary, a sample of data, and so on.

Using our data integration model, we will be able to see that UNIPROT is easily integrated with OMIM, DISGENET, IID, and DRUGCENTRAL by the *gene name*. By integrating with OMIM, we would have more details about genetic phenotypes related to the gene *inhA*, while DISGENET would bring the variants of the genes related to the tuberculosis disease. IID adds information about how a protein related to *inhA* (*Enoyl-[acyl-carrier-protein] reductase [NADH]*) interacts with other proteins. UNIPROT can also be integrated with REACTOME since REACTOME contains a field named *UNIPROT identifier*. Thus, we would have additional information about how the molecules interact in a cell to change the cell or create a specific product; for instance, turn genes on or off.

Additionally, integrating with DRUGCENTRAL would add information about interactions related to the drugs and tuberculosis. The integration with DRUGCENTRAL will allow integration with DRUGBANK, which brings supplementary details on the substance of the drugs and related products. For instance, we will find that *Pretomanid* is a medication for treating tuberculosis. Finally, having the disease type from DISGENET, we could connect with the MONDO ontology and learn about the different types of the disease, such as endocrine, esophageal, ocular, spinal tuberculosis, and others.

5.5 Complexity & Evaluation

Regarding the complexity of operations, according to Alserafi et al. (2016), the equation to calculate the total number of comparisons that need to be performed to find the columns candidate to data integration is

$$comparisons = \left[d \times \frac{d-1}{2} \right] \times m^2 \tag{5.5}$$

where d represents the number of datasets, and m represents the average number of attributes for each dataset. Our nine dataframes (presented in Table 5.1) have 344 attributes in total, considering nine dataframes, then $m = 38$. Thus, we would have to perform about 51984 comparisons among the attributes.

Regarding the evaluation, our previous literature review identified that the related work usually evaluates their approaches based on scalability, execution time, precision, recall, F1, and accuracy. To be able to evaluate our experiments based on precision, recall, F1, and accuracy, we would have to have a gold standard, which indicates the points for integration for the datasets we selected. As we did not find another work that analyzed the same datasets for data integration, we manually mapped the integration

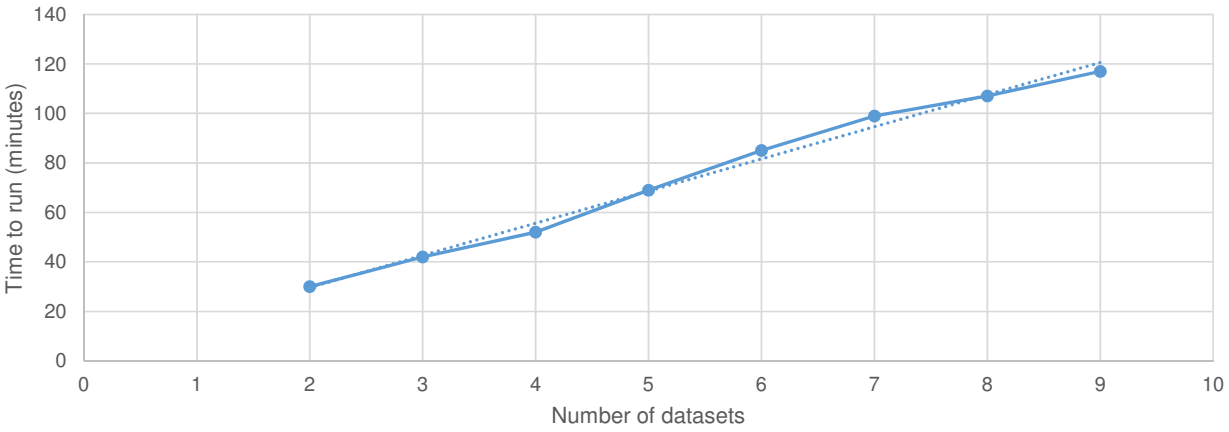


Figure 5.7: Scalability - Execution time
Source: Couto et al. (2022a)

points, as explained before. Therefore, if we calculate those metrics based on our results and the manual mapping, we would have the maximum resulting value for each metric since our model identified all of the manually mapped integration points and even others that were not identified on the manual mapping.

Therefore, we performed an analysis to answer the following question: *1) What is the average execution time and scalability provided by our model, according to the number of dataframes to be analyzed?*. We ran the model 10 times to get the average running time, starting with the bigger dataframe (IID) and ending with the smaller (DRUGBANK) - see Table 5.1 to check the datasets ordered by size. Figure 5.7 shows the results of the scalability evaluation. We started the scalability evaluation with only two dataframes; we added one more dataframe each time and checked the runtime. It takes on average 117 minutes to run for all dataframes in the hardware we described in Subsection 5.1.1. Note that we run it in memory, in hardware with a humble configuration.

The scalability of the proposed solution takes place in terms of enabling comparisons between all attributes of all datasets. For example, the 19 attributes of DRUGCENTRAL are compared with the 253 attributes of the IID and so on, creating a bigger and bigger search space as we add more datasets for comparison.

5.6 Challenges we address in comparison to the state-of-the-art

In this section, we compare our work with the ones we presented in Sections 4.3 and 4.4. In the two SLRs, we presented challenges in big data profiling and in data integration in data lakes. We performed an analysis to check which challenges are already addressed by which papers (see Table 5.3 and Table 5.4). We read the papers searching

for the challenges keywords to perform this investigation. Then we performed an overall reading to check if the papers really do not address the challenges.

In the SLR about big data profiling, we disregard the papers about literature reviews (Abedjan, 2018; Dai et al., 2016; Juddoo, 2015) to compare only the papers that present solutions (frameworks, algorithms, or software) for big data profiling. We also remove the challenge *lack of research* from the Table, since all published papers help to address this challenge. If we recap the results of the SLR about data profiling results, we identify that an optimal solution would address all the 15 challenges. As we explain next, we can see our model creating alternatives to help mitigate them.

1. Complexity: we contribute to this challenge by working with data preparation on diverse types of data.
2. Continuous profiling: using Apache Nifi scheduling process, we can ingest the datasets according to a predefined amount of time and then use HDFS and Python to reprocess the dataframes and rerun the model.
3. Incremental profiling: See *Continuous profiling*.
4. Interpretation: our model allows us to understand and interpret data profiling results by using it for data integration.
5. Lack of research: we contribute by creating a model to help mitigate existing challenges.
6. Metadata: we do not create metadata, but we use existing metadata as input for our model.
7. Online profiling: our model does not directly address this challenge.
8. Poor data quality: we contribute to the field of bioinformatics data quality by preprocessing the datasets to be processed by our model.
9. Profiling dynamic data: See *Continuous profiling*.
10. Topical profiling: we work on the specific topic of bioinformatics datasets.
11. Value: the data integration model we developed can help users make better decisions based on the suggested data integration points.
12. Variability: we work with data that vary regarding size, content, type, and other aspects.
13. Variety: we profile heterogeneous textual data to work on data integration.

Table 5.3: Papers versus challenges they address - RSL Big Data Profiling
Source: Couto et al. (2022b)

Papers	Challenges	Complexity	Continuous profiling	Incremental profiling	Interpretation	Metadata	Online profiling	Poor data quality	Profiling dynamic data	Topical profiling	Value	Variability	Variety	Visualization	Volume
Our work	**13**	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓
Canbek et al. (2018)	9				✓	✓		✓		✓	✓	✓	✓	✓	✓
Ardagna et al. (2018)	7							✓	✓	✓	✓	✓	✓		✓
Chrimes and Zamani (2017)	7	✓			✓	✓				✓	✓			✓	✓
Koehler et al. (2019)	6	✓				✓		✓			✓		✓		✓
Sampaio et al. (2019)	6	✓						✓		✓			✓	✓	✓
Vieira et al. (2020)	6	✓						✓		✓			✓	✓	✓
Santos et al. (2019)	5	✓									✓		✓	✓	✓
Alserafi et al. (2016)	4					✓		✓				✓	✓		
Liu et al. (2013)	4	✓					✓		✓	✓					
Maccioni and Torlone (2017)	4			✓		✓	✓							✓	
Taleb et al. (2019)	4					✓					✓			✓	✓
Khalid and Zimányi (2019)	3	✓				✓								✓	
Jang et al. (2018)	2							✓			✓				
Sun et al. (2018)	2									✓				✓	
Shaabani and Meinel (2018)	1					✓									
Heise et al. (2013)	1													✓	

14. Visualization: our model provides visualization aid to help understand the data integration created by data profiling techniques.

15. Volume: although the datasets we used in our experiments are not quite huge in volume, we believe our model can escalate for bigger datasets if we provide more computational resources than we had available.

Regarding the SLR about data integration in data lakes, we also mapped some related challenges. In the same way in the previous analysis, we omit the challenge *lack of available solutions* from the Table, since all published papers also help in addressing this challenge. Table 5.4 presents the relationships among the challenges and the papers we selected in the second SLR. Next, we discuss how we believe our model contributes to helping mitigating the challenges.

- Complexity: our model ease the complexity of data transformation by ingesting, storing, processing, and visualizing data in an integrated manner.
- Computational cost: we present the experiments that show the computational cost for our model.
- Diversity: our model implements the use of different elements in a data lake, which are easily integrated among them.
- In-memory integration: our model performs in-memory data integration.
- Lack of available solutions: we contribute by creating a model to help mitigate existing challenges.
- Non-generalizable solutions: although we present experiments with bioinformatics datasets, the model can deal with different data domains.
- Scalability: we present the experiments that show the scalability of our model.
- Variability: the user can rerun our model always that the datasets to be analyzed present any changes.
- Variety: our model ingests data from different textual sources and formats.

It is important to note that regarding the papers that are not related to a specific challenge, the authors do not explicitly write about the item - although they also may contribute to the challenge resolution somehow.

By analyzing Tables 5.3 and 5.4, we can see that the challenges complexity, variability, and variety are shared in both areas: big data profiling and big data integration. Although variability and variety are part of the V’s from big data, we can see that there

Table 5.4: Papers versus challenges they address - RSL Data integration in Data Lakes
Source: Couto and Ruiz (2022)

Papers	**Challenges**	Complexity	Computational cost	Diversity	In-memory integration	Non-generalizable solutions	Scalability	Variability	Variety
Our work	8	✓	✓	✓	✓	✓	✓	✓	✓
Jovanovic et al. (2021)	6	✓	✓		✓	✓		✓	✓
Kathiravelu and Sharma (2017)	5	✓		✓	✓		✓		✓
Dong et al. (2021)	5	✓	✓		✓	✓	✓		
Endris et al. (2019)	5	✓	✓		✓	✓			✓
Alrehamy and Walker (2018)	5	✓				✓	✓	✓	✓
Zhu et al. (2019)	5	✓	✓		✓	✓	✓		
Hai et al. (2018)	4	✓		✓		✓	✓		
Alserafi et al. (2016)	4			✓		✓		✓	✓
Yang et al. (2020)	3		✓			✓	✓		
Quinn et al. (2020)	3	✓	✓				✓		
Dhayne et al. (2021)	3	✓		✓					✓
Pomp et al. (2018b)	2	✓							✓
Brackenbury et al. (2018)	2	✓					✓		
Zhang and Ives (2019)	2			✓		✓			
Dabbèchi et al. (2020)	2			✓					✓
Rezig et al. (2021)	2	✓				✓			
Helal et al. (2021)	1			✓					
Beyan et al. (2016)	1								✓
Koutras (2019)	1					✓			
Alili et al. (2017)	1		✓						
Haller and Lenz (2020)	1								✓
Hai and Quix (2019)	1	✓							

are still opportunities to improve in those areas. On the other hand, there are challenges specific to big data profiling, such as *continuous profiling*, *incremental profiling*, *profiling dynamic data*, and *topical profiling*. Likewise, some challenges are more specific to data integration, such as *in-memory integration*.

Remarks

In this chapter, we presented and discussed the architecture and the processes that compose our model for automated data integration in a Hadoop data lake, our data integration algorithm, and we present experiments with eight well-known heterogeneous datasets from bioinformatics domain. We tested the similarity among the dataframes with different similarity measures, and we identified that The Overlap coefficient and Jaccard would be enough for us to validate our proposal. Because the Overlap coefficient presented better results than a specialist analysis, our experiments suggest that the Overlap coefficient is the best option for the in-memory set similarity approach we developed. Based on the Overlap coefficient, we found the top-k overlap set similarity that can help define data integration points for dataframes in a data lake. Moreover, we presented a use case to exemplify the usefulness of the model, alongside an evaluation. To conclude, we present how our work is positioned among other state-of-the-art work, regarding the challenges we approach.

6. CONCLUSIONS

In the introduction of our work, we stated that our Primary Research Question is: *How to create a model to automatize data integration for heterogeneous datasets taking advantage of the Hadoop-based data lake ecosystem?* During the development of this thesis, we discuss our results that indicate that our research points to the answer to that question. In summary, to answer the question, we developed a model based on challenges identified in the scientific literature, which encompass research opportunities in big data profiling and data integration in data lakes. Our model was developed and validated with real data from the field of bioinformatics. Regarding replicability outside Apache Hadoop, while there is a feeling that our solution transcends the framework, this needs to be further investigated to have more conclusive evidence that it would really achieve the same results in another big data ecosystem configuration.

Throughout the development of this thesis, we present the importance of studying the topic of data integration in data lakes and how this topic has been developing in recent years. We present the process and results for the research we carried out over the last four years so that we can contribute to Computer Science, and more specifically, to the big data community. Our main objective for this work was *to develop a model to automatically integrate heterogeneous datasets in data lakes, taking advantage of the Apache Hadoop framework and its ecosystem*, and the main objective was achieved through the specific objectives that we summarize next.

Regarding our specific objectives, Chapter 3 helps us achieve objective 1 (*Identify main concepts related to big data, data lakes, data profiling, data integration, and bioinformatics*). In Chapter 4, we achieved objective 2 (*Identify the main system architectures to create data lakes*) in Section 4.2, objective 3 (*Identify the state-of-the-art and challenges related to big data profiling*) in Section 4.3, and objective 4 (*Identify the state-of-the-art and challenges related to data integration in data lakes*) in Section 4.4. Chapter 5 references objective 5, where we *Create a model for automatized heterogeneous data integration of datasets in data lakes*, *Perform experiments using the model*, *Evaluate the model* (objective 6), and *Compare the model against state-of-the-art implementations* (objective 7). Finally, in the current Chapter, Section 6.3 presents how we *Report results to the scientific community* (objective 8).

6.1 Summary of the Contributions

We develop this thesis to benefit the scientific community (academia) and industry. In this section, we discuss how we contribute with both. Our contributions start with

the results we achieved with the mapping study and systematic literature reviews that permeate the theoretical background and related work.

With the first MS about data lakes, we identify the **most common definitions of the term data lake**, and we map the **most used system architectures in data lake ecosystems**. We think it is helpful for people who want to start learning about data lakes and as a guide to help choose the tools to compose a data lake ecosystem. We also create a **definition for the term data lake** that has been adopted by the academia (until now, cited by: Sawadogo and Darmont (2020); Brous et al. (2020); Joaquim and Mello (2020); Hai et al. (2021); Vitali et al. (2021); Francia et al. (2021); Zhao (2021); Bimonte et al. (2022)). So far, that paper have been read on ResearchGate more than one thousand times[1].

We also reviewed the literature to find the research efforts related to big data profiling. We present a **summary of state of the art related to big data profiling**, and we believe it could be helpful for the scientific community. We also discuss the scenarios and datasets, what metadata is presented in the papers, and what information people extract when using data profiling in big data. Most importantly, we **map the main challenges related to big data profiling**, which can open new research directions related to theory and practice.

With the last SLR, we deepen the knowledge about **the models for data integration in data lakes**. We also identified the most used *similarity metrics for data integration in data lakes*, and we discussed the *evaluation of data integration models for data lakes* and the *type of data they most used in the experiments*. Lastly, we classified the *challenges in data integration in data lakes*. That SLR was useful to define where we could focus our work. As in the MS, we expect the SLRs results to be useful in industry and academia by providing beginners with relevant aspects concerning big data profiling and integration.

The literature reviews served the purpose of basing the decisions for our main contribution: **a model to automatize data integration for heterogeneous datasets taking advantage of the Hadoop-based data lake ecosystem**. We made the code and results available for easing reproducibility, which can be useful for both academia and industry to use our work for future comparisons or as a baseline for new implementations.

6.2 Limitations

We had some limitations during the development of our work. Next, we mention some challenges and how we dealt with them.

[1]Available at https://www.researchgate.net/publication/335150494_A_Mapping_Study_about_Data_Lakes_An_Improved_Definition_and_Possible_Architectures/stats. Accessed in December, 2021.

The first challenge is related to data volume. Big data is usually represented by petabytes of data. Nowadays, in our laboratory, we do not have the infrastructure to create a cluster to support this data size. To overcome this challenge, we submitted a proposal to a call [2] opened by CNPq/AWS (*Conselho Nacional de Desenvolvimento Científico e Tecnológico* / Amazon Web Service), to get cloud credits for research. Unfortunately, our project was not selected, and that option rested frustrated. Thus, we decided to use Docker containers to perform the experiments. Although using containers may compromise performance tests because the computational resources are shared instead of really distributed in a cluster, we achieved an almost linear time to run our experiments (see Figure 5.7). We believe that if we had a real cluster of separated nodes, we could have achieved even better performance results.

As a plan B, we imagined we could use the infrastructure provided by Newcastle University, in the inter-university exchange doctorate, to perform the tests. We started the experiments there, and the available hardware was better but not much different from the infrastructure we have in Brazil (it was a shared server with processor Intel® Xeon(R) CPU X5670 @ 2.93GHz × 24, 32 GB RAM, 4 TB disk). Thus, although the computational power we had access lacks robustness, the infrastructure can be representative for some scenarios such as the one we selected for the experiments.

Variety is also not trivial to simulate. Different types of data populate data lakes. Thus, we faced the challenge of selecting different types of data to test our model. Another challenge is related to access to real data, which is not easy to get, mostly when we talk about big data. Additionally, Brazil's recent General Law about Data Protection (LGPD - Law 3.709/2018) and the General Data Protection Regulation (EU GDPR) create some barriers to researchers accessing real data.

To overcome the challenges related to volume, variety, and data access, we decided to select eight well-known datasets, which are freely available for download over the internet. Although they do not represent real big data in volume, they represent well other big data V's, such as the variety in a specific domain (bioinformatics), the variability (they change over time, adding more information), and the veracity and validity (the selected datasets are curated).

A final significant challenge we faced regards the diversity of tools that can be used to compose the Hadoop Ecosystem, having a huge variety of possible combinations. To deal with this issue, we started our research by reviewing the literature about tools to create data lakes, then we analyzed the available options and decided to approach this issue in a more canonical way, selecting among the most used tools the ones we had already worked with, and the ones that we think was essential to help us achieve our main objective.

[2] Available at: Call CNPq/AWS Nº 032/2019. Accessed February, 2020

6.3 Publications

During the doctorate, we developed some studies, mostly related to the present work and some collaborations. Table 6.1 presents a summary of the scientific work we produced since 2019. Papers Couto et al. (2019, 2022b); Couto and Ruiz (2022); Couto et al. (2022a) are entire chapters and sections of this thesis, where we published the three literature reviews and the outline for our model. On Miehe et al. (2019) and Paludo Licks et al. (2020), we developed techniques for automatically select indexes for a database, first using genetic algorithms and then applying reinforcement learning. In Engelmann et al. (2019) we modeled an ontology that can deal with multivariate data. In Borges et al. (2020, 2021) we studied the use of machine learning in the software engineering domain. In Oliveira et al. (2021) we worked on accessible guidelines for ambient intelligence systems. In Couto et al. (2021, 2022c) we explore the results of my master dissertation, related to data visualization in software project management. Finally, in Couto et al. (2022d) we worked with machine learning and deep learning techniques to suggest how to solve new incidents based on the historical data.

6.4 Availability of data and materials

The data that support the findings of this study are available from:

- UNIPROT (UniProtKB - Reviewed (Swiss-Prot)). Available at UniProt Consortium (2021). Release: 2020_06.
- OMIM (genemap2). Available at McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins University (Baltimore, MD) (2021), upon register and request. Release: File generated on 02/07/2020.
- DISGENET: Available at Integrative Biomedical Informatics Group (2021). Release: version 7.0, January 2020.
- DRUGCENTRAL: Available at Avram et al. (2021b). Release: 18/09/2020.
- IID: Available at Kotlyar et al. (2021). Release: 2018-11.
- MONDO: Available at OBO Foundry (2021). Release: v2021-01-15.
- REACTOME: Available at Reactome (2021). Release: Version 75, 07/12/2020.
- DRUGBANK: Available at OMx Personal Health Analytics, Inc. (2021), upon registration. Release: 5.1.8, 2021-01-03.

Table 6.1: Scientific production since 2019
Source: The author (2022)

Year	Title	paper	Type	Conference or Journal Name	Correlation with the thesis	Qualis
2019	A Mapping Study about Data Lakes: An Improved Definition and Possible Architectures	Couto et al. (2019)	Conference Proceedings	International Conference on Software Engineering and Knowledge Engineering	Section 4.2.	B1
2019	Gadis: A Genetic Algorithm for Database Index Selection	Miehe et al. (2019)	Conference Proceedings	International Conference on Software Engineering and Knowledge Engineering	we used artificial intelligence algorithms for selecting indexes to accelerate search in databases.	B1
2019	Towards an Ontology to Support Decision-making in Hospital Bed Allocation	Engelmann et al. (2019)	Conference Proceedings	International Conference on Software Engineering and Knowledge Engineering	We modeled hospital processes, which can generate multivariate data.	B1
2020	How Machine Learning has been applied in Software Engineering?	Borges et al. (2020)	Conference Proceedings	International Conference on Enterprise Information Systems	We identified the most used tools and techniques to ease the software engineering process through machine learning.	A3
2020	SmartIX: A database indexing agent based on reinforcement learning	Paludo Licks et al. (2020)	Journal	Applied Intelligence Journal	We used artificial intelligence algorithms for selecting indexes to accelerate search in databases.	A2
2021	Challenges in using Machine Learning to support Software Engineering	Borges et al. (2021)	Conference Proceedings	International Conference on Enterprise Information Systems	We identified the challenges people face when using machine learning in the software engineering process.	A3
2021	A PMBoK Extension Proposal for Data Visualization in Software Project Management	Couto et al. (2021)	Conference Proceedings	International Conference on Enterprise Information Systems	Results from master thesis (not related).	A3
2021	Improving the Design of Ambient Intelligence Systems: Guidelines based on a Systematic Review	Oliveira et al. (2021)	Journal	International Journal of Human-Computer Interaction	Guidelines for system development.	A2
2022	On the use of Machine Learning and Deep Learning for Text Similarity and Categorization and its Application to Troubleshooting Automation	Couto et al. (2022d)	Conference Proceedings	Hawaii International Conference on System Sciences	We studied text similarity techniques and its application to a specific dataset.	A1
2022	New trends in big data profiling	Couto et al. (2022b)	Conference Proceedings	Computing Conference (formerly called Science and Information Conference)	Section 4.3.	A4
2022	Automatized bioinformatics data integration in a Hadoop-based data lake	Couto et al. (2022a)	Conference Proceedings	International Conference on Data Mining and Applications	Chapter 5.	A4
2022	Extending the Project Management Body of Knowledge (PMBOK) for Data Visualization in Software Project Management	Couto et al. (2022c)	Journal	Springer Nature Computer Science	Results from master thesis (not related).	Non-indexed yet
2022	ML@SE: What do we know about how Machine Learning impact Software Engineering practice?	Borges et al. (2022)	Conference Proceedings	Iberian Conference on Information Systems and Technologies	We identified how to use machine learning in the software engineering practice.	A4
2022	An overview about data integration in data lakes	Couto and Ruiz (2022)	Conference Proceedings	Iberian Conference on Information Systems and Technologies	Section 4.4.	A4

The code we developed is available at: https://github.com/juliacolleoni/DataLake.

Restrictions apply to the availability of some of these data, which were used under license for the current study, and so are not all publicly available. Data are, however, available from the author upon reasonable request and with permission of the owners, when necessary.

6.5 Future work

For future work, we plan to develop a few extensions for our model:

- Implement text similarity strategies to magnify the reach of our results and increase the points for data integration based on textual semantics and syntactic.
- Create an integrated environment using Neo4J to allow queries over the integrated dataframes.
- Perform experiments with big data benchmarks and other domains of datasets.
- Perform experiments with other tools available in the Hadoop ecosystem, such as Apache Kafka for data ingestion, MongoDB for data storage, and Apache Spark for data processing.
- Improve data security in the data lake, providing authentication and authorization processes.

6.6 Final Remarks

This Chapter completes this thesis, presenting the conclusions of our work and how we achieve our general and specific objectives, alongside a synopsis of our main contributions and limitations. We present the work we have published during the doctoral period and where to find the data and code to reproduce our results. Finally, we present an outline of future research that could be derived from this thesis.

REFERENCES

Abadi, D., Ailamaki, A., Andersen, D., Bailis, P., Balazinska, M., Bernstein, P., ..., and Suciu, D. (Dec, 2019). The seattle report on database research. *SIGMOD Record*, vol. 48, pp. 44–53.

Abedjan, Z. (2018). An introduction to data profiling. In: *Business Intelligence and Big Data*, pp. 1–20, Cham, DE. Springer.

Abedjan, Z., Golab, L., and Naumann, F. (2016). Data profiling. In: *International Conference on Data Engineering*, pp. 1432–1435, Helsinki, FI. IEEE.

Abedjan, Z., Golab, L., and Naumann, F. (2017). Data profiling: A tutorial. In: *International Conference on Management of Data*, pp. 1747–1751, Chicago, US. ACM.

Abedjan, Z., Golab, L., and Naumann, F. (Jun, 2015). Profiling relational data: a survey. *The Very Large Databases Journal*, vol. 24, pp. 557–581.

Ahmad, F., Sarkar, A., and Debnath, N. C. (2017). QoS lake: Challenges, design and technologies. In: *International Conference on Recent Advances in Signal Processing, Telecommunications and Computing*, pp. 65–70, Da Nang, VN. IEEE.

Ahmadov, A., Thiele, M., Eberius, J., Lehner, W., and Wrembel, R. (2015). Towards a hybrid imputation approach using web tables. In: *International Symposium on Big Data Computing*, pp. 21–30, Chennai, IN. IEEE/ACM.

Alili, H., Belhajjame, K., Drira, R., Grigori, D., and Ghézala, H. H. B. (2018). Quality based data integration for enriching user data sources in service lakes. In: *International Conference on Web Services*, pp. 163–170, San Francisco, US. IEEE.

Alili, H., Belhajjame, K., Grigori, D., Drira, R., and Ben Ghezala, H. (Jun, 2017). On enriching user-centered data integration schemas in service lakes. *Lecture Notes in Business Information Processing*, vol. 288, pp. 3–15.

Alrehamy, H. and Walker, C. (Mar, 2018). SemLinker: automating big data integration for casual users. *Journal of Big Data*, vol. 5, pp. 1–14.

Alserafi, A., Abelló, A., Romero, O., and Calders, T. (2016). Towards information profiling: Data lake content metadata management. In: *International Conference on Data Mining Workshops*, pp. 178–185, Barcelona, ES. IEEE.

Amghar, S., Cherdal, S., and Mouline, S. (May, 2020). Storing, preprocessing and analyzing tweets: finding the suitable nosql system. *International Journal of Computers and Applications*, vol. 44, pp. 1–10.

Apache, S. F. (2019). Apache Hadoop. Retrieved from https://hadoop.apache.org. November, 2019.

Ardagna, D., Cappiello, C., Samá, W., and Vitali, M. (Dec, 2018). Context-aware data quality assessment for big data. *Future Generation Computer Systems*, vol. 89, pp. 548 – 562.

Auer, S., Scerri, S., Versteden, A., Pauwels, E., Charalambidis, A., Konstantopoulos, S., ..., and Vidal, M.-E. (2017). The BigDataEurope platform – supporting the variety dimension of big data. In: *International Conference on Web Engineering*, pp. 41–59, Rome, IT. Springer.

Avram, S., Bologa, C. G., Holmes, J., Bocci, G., Wilson, T. B., Nguyen, D.-T., ..., and Oprea, T. I. (Jan, 2021a). Drugcentral 2021 supports drug discovery and repositioning. *Nucleic Acids Research*, vol. 49, pp. D1160–D1169.

Avram, S., Bologa, C. G., Holmes, J., Bocci, G., Wilson, T. B., Nguyen, D.-T., Curpan, R., Halip, L., Bora, A., Yang, J. J., et al. (2021b). DrugCentral dataset. Retrieved from https://drugcentral.org/download. June, 2021.

Beheshti, A., Benatallah, B., Nouri, R., Chhieng, V. M., Xiong, H., and Zhao, X. (2017). CoreDB: A data lake service. In: *Conference on Information and Knowledge Management*, pp. 2451–2454, Singapore, SG. ACM.

Benaissa, R., Benhammadi, F., Boussaid, O., and Mokhtari, A. (2018). Clustering approach for data lake based on medoid's ranking strategy. In: *International Conference on Computer Science and its Applications*, pp. 250–260, San Francisco, US. Springer.

Beyan, O. D., Handschuh, S., Koumpis, A., Fragidis, G., and Decker, S. (2016). A Framework for Applying Data Integration and Curation Pipelines to Support Integration of Migrants and Refugees in Europe. In: *Working Conference on Virtual Enterprises*, pp. 588–596, Porto, PT. HAL.

Bhandarkar, M. (2017). AdBench: A complete benchmark for modern data pipelines. In: *Technology Conference on Performance Evaluation and Benchmarking*, pp. 107–120, Rio de Janeiro, BR. Springer.

Bimonte, S., Gallinucci, E., Marcel, P., and Rizzi, S. (Feb, 2022). Data variety, come as you are in multi-model data warehouses. *Information Systems*, vol. 104, pp. 101734.

Borges, O., Couto, J., Ruiz, D., and Prikladnicki, R. (2020). How machine learning has been applied in software engineering? In: *International Conference on Enterprise Information Systems*, pp. 306–313, Prague, CZ. SCITEPRESS.

Borges, O., Couto, J., Ruiz, D., and Prikladnicki, R. (2021). Challenges in using machine learning to support software engineering. In: *International Conference on Enterprise Information Systems*, pp. 224–231, Online. SCITEPRESS.

Borges, O., Lima, M., Couto, J., Gadelha, B., Conte, T., and Prikladnicki, R. (2022). ML@SE: What do we know about how machine learning impact software engineering practice? In: *Iberian Conference on Information Systems and Technologies*, pp. 6, Madrid, ES. IEEE.

Brackenbury, W., Liu, R., Mondal, M., Elmore, A. J., Ur, B., Chard, K., and Franklin, M. J. (2018). Draining the data swamp: A similarity-based approach. In: *Workshop on Human-In-the-Loop Data Analytics*, pp. 1–7, Houston, US. ACM.

Brereton, P., Kitchenham, B. A., Budgen, D., Turner, M., and Khalil, M. (Apr, 2007). Lessons from applying the systematic literature review process within the software engineering domain. *Journal of Systems and Software*, vol. 80, pp. 571–583.

Brous, P., Janssen, M., and Krans, R. (Mar, 2020). Data governance as success factor for data science. *Responsible Design, Implementation and Use of Information and Communication Technology*, vol. 12066, pp. 431.

Brown, B., Bughin, J., Byers, A. H., Chui, M., Dobbs, R., Manyika, J., and Roxburgh, C. (2011). Big data: The next frontier for innovation, competition, and productivity. Technical report, McKinsey Global Institute.

Canbek, G., Sagiroglu, S., and Taskaya Temizel, T. (2018). New techniques in profiling big datasets for machine learning with a concise review of Android mobile malware datasets. In: *International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism*, pp. 117–121, Ankara, TR. IEEE.

Ceravolo, P., Azzini, A., Angelini, M., Catarci, T., Cudré-Mauroux, P., Damiani, E., ..., and Zaraket, F. (Jun, 2018). Big data semantics. *Journal on Data Semantics*, vol. 7, pp. 65–85.

Cha, B., Park, S., Kim, J., Pan, S. B., and Shin, J. (Feb, 2018). International network performance and security testing based on distributed Abyss storage cluster and draft of data lake framework. *Security and Communication Networks*, vol. 2018, pp. 1–14.

Chen, H., Grider, G., and Montoya, D. R. (2017). An early functional and performance experiment of the MarFS hybrid storage ecosystem. In: *International Conference on Cloud Engineering*, pp. 59–66, Vancouver, CA. IEEE.

Chen, Y., Chen, H., and Huang, P. (2018). Enhancing the data privacy for public data lakes. In: *International Conference on Applied System Invention*, pp. 1065–1068, Chiba and Tokyo, JP. IEEE.

Chrimes, D. and Zamani, H. (Dec, 2017). Using distributed data over HBase in big data analytics platform for clinical services. *Computational and Mathematical Methods in Medicine*, vol. 2017, pp. 1–16.

Ciociola, A., Cocca, M., Giordano, D., Mellia, M., Morichetta, A., Putina, A., and Salutari, F. (2017). UMAP: Urban mobility analysis platform to harvest car sharing data. In: *Smart World Congress*, pp. 1–8, San Francisco, US. IEEE.

Consortium, U. (Jan, 2019). Uniprot: a worldwide hub of protein knowledge. *Nucleic Acids Research*, vol. 47, pp. D506–D515.

Corbett, J. C., Dean, J., Epstein, M., Fikes, A., Frost, C., Furman, J., and Wang, . R. (2012). Spanner: Google's globally-distributed database. In: *Conference on Operating Systems Design and Implementation*, pp. 251–264, Hollywood, US. USENIX Association.

Couto, J., Borges, O., and Ruiz, D. (2022a). Automatized bioinformatics data integration in a hadoop-based data lake. In: *International Conference on Data Mining and Applications*, pp. 16, Vancouver, CA. AIRCC Pulishing Corporation.

Couto, J., Borges, O. T., Ruiz, D., Marczak, S., and Prikladnicki, R. (2019). A mapping study about data lakes: An improved definition and possible architectures. In: *International Conference on Software Engineering and Knowledge Engineering*, pp. 453–458, Lisbon, PT. KSI Research Inc.

Couto, J., Damasio, J., Bordini, R., and Ruiz, D. (2022b). New trends in big data profiling. In: *Computing Conference*, pp. 14, London, UK. Springer.

Couto, J., Kroll, J., Ruiz, D., and Prikladnicki, R. (2021). A pmbok extension proposal for data visualization in software project management. In: *International Conference on Enterprise Information Systems*, pp. 54–65, Online. SCITEPRESS.

Couto, J., Kroll, J., Ruiz, D., and Prikladnicki, R. (Mar, 2022c). Extending the project management body of knowledge (pmbok) for data visualization in software project management. *Springer Nature Computer Science*, vol. 0, pp. 27.

Couto, J. and Ruiz, D. (2022). An overview about data integration in data lakes. In: *Iberian Conference on Information Systems and Technologies*, pp. 6, Madrid, ES. IEEE.

Couto, J., Tomaz, L., Godoy, J., Kniest, D., Callegari, D., Meneguzzi, F., and Ruiz, D. (2022d). On the use of machine learning and deep learning for text similarity and categorization and its application to troubleshooting automation. In: *Hawaii International Conference on System Sciences*, pp. 778–787, Hawaii, US. ScholarSpace.

Dabbèchi, H., Haddar, N. Z., Elghazel, H., and Haddar, K. (2020). Social media data integration: From data lake to nosql data warehouse. In: *International Conference on Intelligent Systems Design and Applications*, pp. 701–710, Online. Springer.

Dai, W., Wardlaw, I., Cui, Y., Mehdi, K., Li, Y., and Long, J. (2016). Data profiling technology of data governance regarding big data: Review and rethinking. In: *Information Technology: New Generations*, pp. 439–450, Cham, DE. Springer.

Dean, J. and Ghemawat, S. (2004). Mapreduce: Simplified data processing on large clusters. In: *Symposium on Operating Systems Design & Implementation*, pp. 10–10, San Francisco, US. USENIX Association.

Dessì, N., Garau, G., Recupero, D. R., and Pes, B. (2016). Increasing open government data transparency with spatial dimension. In: *International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pp. 247–249, Paris, FR. IEEE.

Dhayne, H., Kilany, R., Haque, R., and Taher, Y. (Jun, 2021). Emr2vec: Bridging the gap between patient data and clinical trial. *Computers & Industrial Engineering*, vol. 156, pp. 107236.

Dholakia, A., Venkatachar, P., Doshi, K., Durgavajhala, R., Tate, S., Schiefer, B., ..., and Sagar, R. S. (2017). Designing a high performance cluster for large-scale SQL-on-Hadoop analytics. In: *International Conference on Big Data*, pp. 1701–1703, Boston, US. IEEE.

Diamantini, C., Giudice, P. L., Musarella, L., Potena, D., Storti, E., and Ursino, D. (2018). An approach to extracting thematic views from highly heterogeneous sources of a data lake. In: *Italian Symposium on Advanced Database Systems*, pp. 1–12, Ginosa, IT. SISINF lab.

Dixon, J. (2010). Pentaho, Hadoop, and data lakes. Retrieved from https://jamesdixon.wordpress.com/2010/10/14. November, 2019.

Dong, Y., Takeoka, K., Xiao, C., and Oyamada, M. (2021). Efficient joinable table discovery in data lakes: A high-dimensional similarity-based approach. In: *International Conference on Data Engineering*, pp. 456–467, Chania, GR. IEEE.

Dutta, H. (Dec, 2015). Graph based data governance model for real time data ingestion. *Computer Society of India Transactions on Information and Communication Technology*, vol. 3, pp. 119–125.

Endris, K. M., Rohde, P. D., Vidal, M.-E., and Auer, S. (2019). Ontario: Federated query processing against a semantic data lake. In: *International Conference on Database and Expert Systems Applications*, vol. 11706, pp. 379–395, Bratislava, SK. Springer.

Engelmann, D., Couto, J., Gabriel, V., Vieira, R., and Bordini, R. (2019). Towards an ontology to support decision-making in hospital bed allocation. In: *International Conference on Software Engineering and Knowledge Engineering*, pp. 71–74, Lisbon, PT. KSI Research Inc.

Europe, B. D. (2021). Apache hadoop docker image. Retrieved from https://github.com/big-data-europe/docker-hadoopl. November, 2021.

Fang, H. (2015). Managing data lakes in big data era: What's a data lake and why has it became popular in data management ecosystem. In: *International Conference on Cyber Technology in Automation, Control, and Intelligent Systems*, pp. 820–824, Shenyang, CN. IEEE.

Farid, M., Roatis, A., Ilyas, I. F., Hoffmann, H.-F., and Chu, X. (2016). CLAMS: Bringing quality to data lakes. In: *International Conference on Management of Data*, pp. 2089–2092, San Francisco, US. ACM.

Farrugia, A., Claxton, R., and Thompson, S. (2016). Towards social network analytics for understanding and managing enterprise data lakes. In: *International Conference on Advances in Social Networks Analysis and Mining*, pp. 1213–1220, San Francisco, US. IEEE/ACM.

Forster, R. R. (2018). *Hive on Spark and MapReduce: A methodology for parameter tuning*. Master Thesis, NOVA Information Management School, Lisbon, PT.

Francia, M., Gallinucci, E., Golfarelli, M., Leoni, A. G., Rizzi, S., and Santolini, N. (Dec, 2021). Making data platforms smarter with moses. *Future Generation Computer Systems*, vol. 125, pp. 299–313.

Gao, Y., Huang, S., and Parameswaran, A. G. (Jun, 2017). Navigating the data lake with Datamaran. *Computing Research Repository - arXiv*, vol. abs/1708.08905, pp. 1–20.

García, I., Martínez-Prieto, M. A., Bregón, A., Álvarez, P. C., and Díaz, F. (2017). Towards a scalable architecture for flight data management. In: *International Conference on Data Science, Technology and Applications*, pp. 263–268, Madrid, ES. SciTePress.

Ghemawat, S., Gobioff, H., and Leung, S.-T. (2003). The Google file system. In: *Symposium on Operating Systems Principles*, pp. 29–43, Bolton Landing, US. ACM.

Gollapudi, S. (2015). Aggregating financial services data without assumptions. In: *International Conference on Semantic Computing*, pp. 312–315, Anaheim, US. IEEE.

Golov, N. and Rönnbäck, L. (Nov, 2017). Big data normalization for massively parallel processing databases. *Computer Standards & Interfaces*, vol. 54, pp. 86 – 93.

Gröger, C. (Mar, 2018). Building an industry 4.0 analytics platform. *Datenbank-Spektrum*, vol. 18, pp. 5–14.

Gupta, M., Patwa, F., and Sandhu, R. (2018). An attribute-based access control model for secure big data processing in Hadoop ecosystem. In: *Workshop on Attribute-Based Access Control*, pp. 13–24, Tempe, US. ACM.

Hai, R., Geisler, S., and Quix, C. (2016). Constance: An intelligent data lake system. In: *International Conference on Management of Data*, pp. 2097–2100, San Francisco, US. ACM.

Hai, R. and Quix, C. (Jul, 2019). Rewriting of plain so tgds into nested tgds. *Very Large Databases Endowment*, vol. 12, pp. 1526–1538.

Hai, R., Quix, C., and Jarke, M. (Jun, 2021). Data lake concept and systems: a survey. *Computing Research Repository - arXiv*, vol. abs/2106.09592, pp. 1–25.

Hai, R., Quix, C., and Zhou, C. (2018). Query rewriting for heterogeneous data lakes. In: *European Conference on Advances in Databases and Information Systems*, pp. 35–49, Budapest, HU. Springer.

Halevy, A. Y., Korn, F., Noy, N. F., Olston, C., Polyzotis, N., Roy, S., and Whang, S. E. (Mar, 2016). Managing Google's data lake: an overview of the goods system. *IEEE Data Engineering Bulletin*, vol. 39, pp. 5–14.

Haller, D. and Lenz, R. (2020). Pharos: Query-driven schema inference for the semantic web. In: *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pp. 112–124, Wurzburg, GE. Springer.

Heise, A., Quiané-Ruiz, J., Abedjan, Z., Jentzsch, A., and Naumann, F. (Dec, 2013). Scalable discovery of unique column combinations. *Very Large Databases Endowment*, vol. 7, pp. 301–312.

Helal, A., Helali, M., Ammar, K., and Mansour, E. (Jul, 2021). A demonstration of kglac: A data discovery and enrichment platform for data science. *Very Large Databases Endowment*, vol. 14, pp. 2675–2678.

Hendler, J. (Dec, 2014). Data integration for heterogenous datasets. *Big Data*, vol. 2, pp. 205–215.

Herman, J., Herman, H., Mathews, M. J., and Vosloo, J. C. (Oct, 2018). Using big data for insights into sustainable energy consumption in industrial and mining sectors. *Journal of Cleaner Production*, vol. 197, pp. 1352 – 1364.

Himmelstein, D. S. (2016). User-friendly extensions of the DrugBank database v1.0. Retrieved from https://doi.org/10.5281/zenodo.45579. November, 2021.

Hui, J., Li, L., and Zhang, Z. (2018). Integration of big data: A survey. In: *International Conference of Pioneering Computer Scientists, Engineers and Educators*, pp. 101–121, Zhengzhou, CN. Springer.

Integrative Biomedical Informatics Group (2021). DisGeNET curated gene-disease associations dataset. Retrieved from https://www.disgenet.org/static/disgenet_ap1/files/downloads/curated_gene_disease_associations.tsv.gz. June, 2021.

ISO/IEC/IEEE (2010). ISO/IEC/IEEE 24765 International Standard - Systems and software engineering – Vocabulary. Standard, ISO/IEC/IEEE.

Jaccard, P. (Jan, 1902). Distribution comparée de la flore alpine dans quelques régions des alpes occidentales et orientales. *Bulletin de la Murithienne*, vol. XXXVII, pp. 81–92.

Jang, W.-J., Kim, J.-Y., Lim, B.-T., and Gim, G.-Y. (Mar, 2018). A study on data profiling based on the statistical analysis for big data quality diagnosis. *International Journal of Advanced Science and Technology*, vol. 117, pp. 77–88.

Jarke, M. (2017). Data spaces: Combining goal-driven and data-driven approaches in community decision and negotiation support. In: *International Conference on Group Decision and Negotiation*, pp. 3–14, Stuttgart, DE. Springer.

Jarke, M. and Quix, C. (2017). *On Warehouses, Lakes, and Spaces*, chap. 16, pp. 231–245. Springer, New York, US.

Jassal, B., Matthews, L., Viteri, G., Gong, C., Lorente, P., Fabregat, A., ..., and D'Eustachio, P. (Jan, 2020). The reactome pathway knowledgebase. *Nucleic Acids Research*, vol. 48, pp. D498–D503.

Joaquim, J. L. M. and Mello, R. d. S. (2020). An analysis of confidentiality issues in data lakes. In: *International Conference on Information Integration and Web-based Applications & Services*, pp. 168–177, Chiang Mai, TH. ACM.

Johnson, T. (2009). *Data profiling*, chap. D, pp. 604–608. Springer, Boston, US.

Jovanovic, P., Nadal, S., Romero, O., Abelló, A., and Bilalli, B. (Dec, 2021). Quarry: a user-centered big data integration platform. *Information Systems Frontiers*, vol. 23, pp. 9–33.

Jovanovic, P., Romero, O., and Abelló, A. (2016). *A unified view of data-intensive flows in business intelligence systems: A survey*, chap. 3, pp. 66–107. Springer, Berlin, DE.

Juddoo, S. (2015). Overview of data quality challenges in the context of big data. In: *International Conference on Computing, Communication and Security*, pp. 1–9, Pamplemousses, MU. IEEE.

Karpathiotakis, M., Floratou, A., Özcan, F., and Ailamaki, A. (2017). No data left behind: Real-time insights from a complex data ecosystem. In: *Symposium on Cloud Computing*, pp. 108–120, Santa Clara, US. ACM.

Kasrin, N., Qureshi, M., Steuer, S., and Nicklas, D. (Mar, 2018). Semantic data management for experimental manufacturing technologies. *Datenbank-Spektrum*, vol. 18, pp. 27–37.

Kassner, L., Gröger, C., Königsberger, J., Hoos, E., Kiefer, C., Weber, C., ..., and Mitschang, B. (2017). The Stuttgart IT architecture for manufacturing. In: *International Conference on Enterprise Information Systems*, pp. 53–80, Porto, PT. Springer.

Kathiravelu, P. and Sharma, A. (2017). A dynamic data warehousing platform for creating and accessing biomedical data lakes. In: *Very Large Data Bases Workshop on Data Management and Analytics for Medicine and Healthcare*, pp. 101–120, Munich, DE. Springer.

Khalid, H. and Zimányi, E. (Sep, 2019). Using rule and goal based agents to create metadata profiles. *Communications in Computer and Information Science*, vol. 1064, pp. 365–377.

Khine, P. P. and Wang, Z. S. (Feb, 2018). Data lake: A new ideology in big data era. *Information Technology, Computer Science and Mathematics ITM Web of Conferences*, vol. 17, pp. 1–11.

Koehler, M., Abel, E., Bogatu, A., Civili, C., Mazilu, L., Konstantinou, N., Fernandes, A., Keane, J., Libkin, L., and Paton, N. W. (Apr, 2019). Incorporating data context to cost-effectively automate end-to-end data wrangling. *IEEE Transactions on Big Data*, vol. X, pp. 1–18.

Kondylakis, H., Koumakis, L., Tsiknakis, M., and Marias, K. (2018). Implementing a data management infrastructure for big healthcare data. In: *International Conference on Biomedical and Health Informatics*, pp. 361–364, Tumkuru, IN. IEEE.

Kotlyar, M., Pastrello, C., Malik, Z., and Jurisica, I. (2021). The iid database. Retrieved from http://iid.ophid.utoronto.ca/static/download/human_annotated_PPIs.txt.gz. June, 2021.

Kotlyar, M., Pastrello, C., Malik, Z., and Jurisica, I. (Jan, 2019). Iid 2018 update: context-specific physical protein–protein interactions in human, model organisms and domesticated species. *Nucleic Acids Research*, vol. 47, pp. D581–D589.

Koutras, C. (2019). Data as a language: A novel approach to data integration. In: *International Conference on Very Large Database - PhD Workshop*, pp. 1–4, Los Angeles, US. Springer.

Landis, J. R. and Koch, G. G. (Mar, 1977). The measurement of observer agreement for categorical data. *Biometrics*, vol. 33, pp. 159–174.

Le, T.-H.-Y., Phan, T.-C., and Phan, A.-C. (2018). Big data driven architecture for medical knowledge management systems in intracranial hemorrhage diagnosis. In: *International Symposium on Integrated Uncertainty in Knowledge Modelling and Decision Making*, pp. 214–225, Hanoi, VN. Springer.

Le Meur, J.-Y. and Tarocco, N. (Sep, 2019). The obsolescence of information and information systems CERN digital memory project. *European Physical Journal Web of Conferences*, vol. 214, pp. 1–8.

Lenzerini, M. (2002). Data integration: A theoretical perspective. In: *Symposium on Principles of Database Systems*, pp. 233–246, New York, US. ACM.

Lesk, A. (2019). *Introduction to bioinformatics*. Oxford university press, Oxford.

Levy, A. Y. (2000). *Logic-Based Techniques in Data Integration*, chap. 24, pp. 575–595. Springer, Boston, MA.

Li, C., Cui, C., He, B., Fan, D., Wang, J., Li, S., ..., and Chen, X. (2017). The design and application of astronomy data lake in China-VO. In: *Astronomical Data Analysis Software and Systems Conference*, pp. 157, Santiago de Chile, CL. ASP.

Lin, X., Li, X., and Lin, X. (Mar, 2020). A review on applications of computational methods in drug screening and design. *Molecules*, vol. 25, pp. 1–17.

Liu, B., Chen, H., Sharma, A., Jiang, G., and Xiong, H. (2013). Modeling heterogeneous time series dynamics to profile big sensor data in complex physical systems. In: *International Conference on Big Data*, pp. 631–638, Santa Clara, US. IEEE.

Maccioni, A. and Torlone, R. (Aug, 2017). Crossing the finish line faster when paddling the data lake with KAYAK. *Very Large Databases Endowment*, vol. 10, pp. 1853–1856.

Madden, S. (May-Jun, 2012). From databases to big data. *IEEE Internet Computing*, vol. 16, pp. 4–6.

Madera, C. and Laurent, A. (2016). The next information architecture evolution: The data lake wave. In: *International Conference on Management of Digital EcoSystems*, pp. 174–180, Biarritz, FR. ACM.

Maksymowicz, K. P., Płaczek, A., Jaksik, R., Student, S., Borys, D., Mrozek, D., Fujarewicz, K., and Świerniak, A. (2016). A holistic approach to testing biomedical hypotheses and analysis of biomedical data. In: *International Conference Beyond Databases, Architectures and Structures*, pp. 449–462, Ustroń, PL. Springer.

Martínez-Prieto, M. A., Bregon, A., García-Miranda, I., Álvarez Esteban, P. C., Díaz, F., and Scarlatti, D. (2017). Integrating flight-related information into a (big) data lake. In: *Digital Avionics Systems Conference*, pp. 1–10, Saint Petersburg, US. IEEE/AIAA.

McCarthy, S., McCarren, A., and Roantree, M. (2018). Combining web and enterprise data for lightweight data mart construction. In: *International Conference on Database and Expert Systems Applications*, pp. 138–146, Regensburg, DE. Springer.

McHugh, M. L. (Oct, 2012). Interrater reliability: The Kappa statistic. *Biochemia Medica*, vol. 22, pp. 276–282.

McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins University (Baltimore, MD) (2021). Online mendelian inheritance in man, omim®. Retrieved from https://www.omim.org. June, 2021.

McPadden, J., Durant, T. J. S., Bunch, D. R., Coppi, A., Price, N., Rodgerson, K., ..., and Schulz, W. L. (Aug, 2018). A scalable data science platform for healthcare and precision medicine research. *Computing Research Repository - arXiv*, vol. abs/1808.04849, pp. 1–7.

Meena, S. D. and Meena, S. V. (Oct, 2016). Data lakes - a new repository for big data analytics. *International Journal of Advanced Research in Computer Science*, vol. 7, pp. 65–67.

Miehe, P., Couto, J., Wehrmann, J., Ruiz, D., and Meneguzzi, F. (2019). Gadis: A genetic algorithm for database index selection. In: *International Conference on Software Engineering and Knowledge Engineering*, pp. 39–42, Lisbon, PT. KSI Research Inc.

Miloslavskaya, N. and Tolstoy, A. (Jul, 2016). Big data, fast data and data lake concepts. *Procedia Computer Science*, vol. 88, pp. 300 – 305.

Mitrovic, S. (Oct, 2017). Specifics of the integration of business intelligence and big data technologies in the processes of economic analysis. *Business Informatics*, vol. 42, pp. 40–46.

Moher, D., Shamseer, L., Clarke, M., Ghersi, D., Liberati, A., Petticrew, M., Shekelle, P., and Stewart, L. A. (Jan, 2015). Preferred reporting items for systematic review and meta-analysis protocols (PRISMA-P) 2015 statement. *Systematic Reviews*, vol. 4, pp. 1–9.

Mrozek, B. M., Stabla, M., and Mrozek, D. (Oct, 2018). Soft and declarative fishing of information in big data lake. *IEEE Transactions on Fuzzy Systems*, vol. 26, pp. 2732–2747.

Mungall, C. J., McMurry, J. A., Köhler, S., Balhoff, J. P., Borromeo, C., Brush, M., ..., and Osumi-Sutherland, D. (Jan, 2017). The monarch initiative: an integrative data and analytic platform connecting phenotypes to genotypes across species. *Nucleic Acids Research*, vol. 45, pp. D712–D722.

Munshi, A. A. and Mohamed, Y. A. I. (Jul, 2018). Data lake lambda architecture for smart grids big data analytics. *IEEE Access*, vol. 6, pp. 40463–40471.

Murdoch, U. (2018). Systematic reviews: Using PICO or PICo. Retrieved from https://libguides.murdoch.edu.au/systematic/PICO. November, 2019.

Nogueira, I. D., Romdhane, M., and Darmont, J. (Jul, 2018). Modeling data lake metadata with a data vault. *Computing Research Repository - arXiv*, vol. abs/1807.04035, pp. 1–19.

Noyes, D. (2021). The top 10 valuable Facebook statistics – q2 2021. Retrieved from https://zephoria.com/top-15-valuable-facebook-statistics. December, 2021.

OBO Foundry (2021). Mondo dataset - json edition. Retrieved from http://purl.obolibrary.org/obo/mondo/mondo-with-equivalents.json. June, 2021.

O'Leary, D. E. (Sep-Oct, 2014). Embedding AI and crowdsourcing in the big data lake. *Intelligent Systems*, vol. 29, pp. 70–73.

Oliveira, J. D., Couto, J. C., Paixão-Cortes, V. S. M., and Bordini, R. H. (May, 2021). Improving the design of ambient intelligence systems: Guidelines based on a systematic review. *International Journal of Human–Computer Interaction*, vol. 38, pp. 19–27.

OMx Personal Health Analytics, Inc. (2021). DrugBank dataset. Retrieved from https://go.drugbank.com/releases/latest. June, 2021.

Ontañón, S. (Feb, 2020). An overview of distance and similarity functions for structured data. *Artificial Intelligence Review*, vol. 53, pp. 5309–5351.

Paludo Licks, G., Couto, J., Miehe, P., de Paris, R., Ruiz, D., and Meneguzzi, F. (Aug, 2020). Smartix: A database indexing agent based on reinforcement learning. *Applied Intelligence*, vol. 50, pp. 2575–2588.

Papenbrock, T., Bergmann, T., Finke, M., Zwiener, J., and Naumann, F. (Aug, 2015). Data profiling with Metanome. *Very Large Databases Endowment*, vol. 8, pp. 1860–1863.

Pena, E., Schmutzer, R., Stephan, C., Reinero, C., Milli, J., Guerra, J., and Osorio, J. (2018). Framework to use modern big data software tools to improve operations at the Paranal Observatory. In: *Observatory Operations: Strategies, Processes, and Systems*, pp. 1–11, Austin, US. SPIE.

Petricioli, L., Humski, L., and Vrdoljak, B. (2021). The challenges of nosql data warehousing. In: *E-business Technologies Conference*, pp. 44–48, Belgrade, RS. IEEE.

Piñero, J., Ramírez-Anguita, J. M., Saüch-Pitarch, J., Ronzano, F., Centeno, E., Sanz, F., and Furlong, L. I. (Jan, 2020). The disgenet knowledge platform for disease genomics: 2019 update. *Nucleic Acids Research*, vol. 48, pp. D845–D855.

Pomp, A., Paulus, A., Jeschke, S., and Meisen, T. (2018a). Enabling semantics in enterprises. In: *International Conference on Enterprise Information Systems*, pp. 428–450, Budapest, HU. Springer.

Pomp, A., Paulus, A., Kirmse, A., Kraus, V., and Meisen, T. (Sep, 2018b). Applying semantics to reduce the time to analytics within complex heterogeneous infrastructures. *Technologies*, vol. 6, pp. 29.

py_stringmatching Team, A. G. (2021). py_stringmatching users manual – similarity measures. Retrieved from http://anhaidgroup.github.io/py_stringmatching/v0.2.x/SimilarityMeasure.html. November, 2021.

Quinn, C., Shabestari, A. Z., Misic, T., Gilani, S., Litoiu, M., and McArthur, J. (Oct, 2020). Building automation system - bim integration using a linked data structure. *Automation in Construction*, vol. 118, pp. 16.

Quix, C., Hai, R., and Vatov, I. (2016). GEMMS: A generic and extensible metadata management system for data lakes. In: *International Conference on Advanced Information Systems Engineering*, pp. 129–136, Ljubljana, SI. Springer.

Rajesh, K. and Ramesh, K. (Mar-May, 2016). An introduction to data lake. *I-manager's Journal on Information Technology*, vol. 5, pp. 1–4.

Ramakrishnan, R., Sridharan, B., Douceur, J. R., Kasturi, P., Krishnamachari-Sampath, B., Krishnamoorthy, K., ..., and Venkatesan, R. (2017). Azure data lake store: A hyperscale distributed file service for big data analytics. In: *International Conference on Management of Data*, pp. 51–63, Chicago, US. ACM.

Ramesh, B. (2015). *Big data architecture*, chap. 2, pp. 29–59. Springer, New Delhi, IN.

Rangarajan, S., Liu, H., Wang, H., and Wang, C.-L. (2018). Scalable architecture for personalized healthcare service recommendation using big data lake. In: *Australasian Symposium on Service Research and Innovation*, pp. 65–79, Wollongong and Sydney, AU. Springer.

Reactome (2021). Reactome UniProt to pathways dataset. Retrieved from https://reactome.org/download/current/UniProt2Reactome_All_Levels.txt. June, 2021.

Revathy, P. and Mukesh, R. (2017). Analysis of big data security practices. In: *International Conference on Applied and Theoretical Computing and Communication Technology*, pp. 264–267, Tunkuru, IN. iCATccT.

Rezig, E., Vanterpool, A., Gadepally, V., Price, B., Cafarella, M., and Stonebraker, M. (Sep, 2021). Towards data discovery by example. *Lecture Notes in Computer Science*, vol. 12633, pp. 66–71.

Roman, J. (2022). The Hadoop ecosystem table. Retrieved from https://hadoopecosystemtable.github.io. January, 2022.

Sackett, D. L. (2000). *Evidence-based medicine: How to practice and teach EBM*. Churchill Livingstone, London, UK.

Sagiroglu, S. and Sinanc, D. (2013). Big data: A review. In: *International Conference on Collaboration Technologies and Systems*, pp. 42–47, San Diego, US. ACM/IEEE/IFIP.

Sampaio, S., Aljubairah, M., Permana, H. A., and Sampaio, P. (Mar, 2019). A conceptual approach for supporting traffic data wrangling tasks. *The Computer Journal*, vol. 62, pp. 461–480.

Sankaranarayanan, H. B. and Lalchandani, J. (2017). Passenger reviews reference architecture using big data lakes. In: *International Conference on Cloud Computing, Data Science and Engineering-Confluence*, pp. 204–209, Noida, IN. IEEE.

Santos, M., Costa, C., Galvão, J., Andrade, C., Pastor, O., and Marcén, A. (Jun, 2019). Enhancing big data warehousing for efficient, integrated and advanced analytics: visionary paper. *Lecture Notes in Business Information Processing*, vol. 350, pp. 215–226.

Santos, R. S., Vaz, T. A., Santos, R. P., and de Oliveira, J. M. P. (2016). Big data analytics in a public general hospital. In: *International Conference on Machine Learning, Optimization and Big Data*, pp. 433–441, Volterra, IT. Springer.

Sawadogo, P. and Darmont, J. (Jun, 2020). On data lake architectures and metadata management. *Journal of Intelligent Information Systems*, vol. 56, pp. 97–120.

Searls, D. B. (Jan, 2005). Data integration: challenges for drug discovery. *Nature Reviews Drug Discovery*, vol. 4, pp. 45–58.

Shaabani, N. and Meinel, C. (2018). Improving the efficiency of inclusion dependency detection. In: *International Conference on Information and Knowledge Management*, pp. 207–216, Torino, IT. ACM.

Sharma, S. (Jun, 2016). Expanded cloud plumes hiding big data ecosystem. *Future Generation Computer Systems*, vol. 59, pp. 63 – 92.

Shlyuger, G. (2017). Apply analytical grid processing to sensor data collections. In: *Cyber Sensing*, pp. 13, Anaheim, US. SPIE.

Singh, K., Paneri, K., Pandey, A., Gupta, G., Sharma, G., Agarwal, P., and Shroff, G. (2016). Visual bayesian fusion to navigate a data lake. In: *International Conference on Information Fusion*, pp. 987–994, Heidelberg, DE. ISIF.

Skluzacek, T. J., Chard, K., and Foster, I. (2016). Klimatic: A virtual data lake for harvesting and distribution of geospatial data. In: *International Workshop on Parallel Data Storage and Data Intensive Scalable Computing*, pp. 31–36, Salt Lake City, US. IEEE.

Solar, G. V., Zechinelli-Martini, J. L., and Espinosa-Oviedo, J. A. (Dec, 2017). Big data management: What to keep from the past to face future challenges? *Data Science and Engineering*, vol. 2, pp. 328–345.

Sørensen, T. J. (1948). *A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons*. I kommission hos Ejnar Munksgaard, Copenhagen, DK.

Spangler, T. (2017). Netflix bandwidth usage climbs to nearly 37% of internet traffic at peak hours. Retrieved from https://variety.com/2015/digital/news/netflix-bandwidth-usage-internet-traffic-1201507187. November, 2019.

Sreekala PK, R. K. (Mar, 2018). Data lake in the big data era: An overview. *Library Herald*, vol. 56, pp. 11–15.

Srinivasan, M. K. and Revathy, P. (2018). State-of-the-art big data security taxonomies. In: *Innovations in Software Engineering Conference*, pp. 16:1–16:7, Hyderabad, IN. ACM.

StackOverflow (2019). Annual developer survey results. Retrieved from https://insights.stackoverflow.com/survey/2019. November, 2019.

Statista, R. D. (2021). Facebook: number of monthly active users worldwide 2008-2021. Retrieved from https://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide/. December, 2021.

Stefanowski, J., Krawiec, K., and Wrembel, R. (Dec, 2017). Exploring complex and big data. *International Journal of Applied Mathematics and Computer Science*, vol. 27, pp. 669–679.

Sun, H., Hu, S., McIntosh, S., and Cao, Y. (Feb, 2018). Big data trip classification on the New York City taxi and Uber sensor network. *Journal of Internet Technology*, vol. 19, pp. 591–598.

Suriarachchi, I. and Plale, B. (2016a). Crossing analytics systems: A case for integrated provenance in data lakes. In: *International Conference on e-Science*, pp. 349–354, Baltimore, US. IEEE.

Suriarachchi, I. and Plale, B. (2016b). Provenance as essential infrastructure for data lakes. In: *International Provenance and Annotation Workshop*, pp. 178–182, McLean, US. Springer.

Taher, Y., Haque, R., AlShaer, M., Heuvel, W., Hacid, M.-S., and Dbouk, M. (2016). A context-aware analytics for processing tweets and analysing sentiment in realtime. In: *Confederated International Conferences: On the Move to Meaningful Internet Systems*, pp. 910–917, Rhodes, GR. Springer.

Taher, Y., Haque, R., AlShaer, M., Heuvel, W., Zeitouni, K., Araujo, R., ..., and Dbouk, M. (2017). A service-based system for sentiment analysis and visualization of Twitter data in realtime. In: *International Conference on Service-Oriented Computing*, pp. 199–202, Malaga, ES. Springer.

Taleb, I., Serhani, M., and Dssouli, R. (Jun, 2019). Big data quality: A data quality profiling model. *Lecture Notes in Computer Science*, vol. 11517, pp. 61–77.

Team, A. N. (2021). Apache NiFi Overview. Retrieved from http://nifi.apache.org/docs.html. November, 2021.

Terrizzano, I. G., Schwarz, P. M., Roth, M., and Colino, J. E. (2015). Data wrangling: The challenging journey from the wild to the lake. In: *Conference on Innovative Data Systems Research*, pp. 1–9, California, US. CIDR.

Tovernić, S., Banović, V., Hrastić, Z., Plantić, K., Šandić, A., and Baranović, M. (2018). Solution for detecting sensitive data inside a data lake. In: *International Convention on Information and Communication Technology, Electronics and Microelectronics*, pp. 1284–1288, Opatija, HR. IEEE.

Tversky, A. (Mar, 1977). Features of similarity. *Psychological Review*, vol. 84, pp. 327.

UN (2019). United Nations – World population prospects 2019. Retrieved from https://population.un.org/wpp/Download/Standard/Population. December, 2021.

UniProt Consortium (2021). UniProt KB Reviewed (Swiss-Prot) dataset. Retrieved from https://www.uniprot.org/downloads. June, 2021.

Vieira, A. A., Dias, L. M., Santos, M. Y., Pereira, G. A., and Oliveira, J. A. (Jan, 2020). On the use of simulation as a big data semantic validator for supply chain management. *Simulation Modelling Practice and Theory*, vol. 98, pp. 1–13.

Vijaymeena, M. and Kavitha, K. (Mar, 2016). A survey on similarity measures in text mining. *Machine Learning and Applications: An International Journal*, vol. 3, pp. 19–28.

Villegas-Ch, W., Luján-Mora, S., Buenaño-Fernandez, D., and Palacios-Pacheco, X. (2018). Big data, the next step in the evolution of educational data analysis. In: *International Conference on Information Technology and Systems*, pp. 138–147, Bangkok, TH. Springer.

Vitali, G., Francia, M., Golfarelli, M., and Canavari, M. (Jan, 2021). Crop management with the iot: An interdisciplinary survey. *Agronomy*, vol. 11, pp. 181.

Walker, C. and Alrehamy, H. (2015). Personal data lake with data gravity pull. In: *International Conference on Big Data and Cloud Computing*, pp. 160–167, Jeju Island, KR. ACM.

Wazlawick, R. (2014). *Metodologia de pesquisa para ciência da computação*. Elsevier, Rio de Janeiro, BR.

Wenning, R. and Kirrane, S. (2018). *Compliance using metadata*, chap. 3, pp. 31–45. Springer, New York, US.

White, T. (2015). *Hadoop: The definitive guide*. O'Reilly Media, Inc., Sebastpol, CA.

Wibowo, M., Sulaiman, S., and Shamsuddin, S. M. (2017). Machine learning in data lake for combining data silos. In: *International Conference on Data Mining and Big Data*, pp. 294–306, Fukuoka, JP. Springer.

Wishart, D. S., Knox, C., Guo, A. C., Shrivastava, S., Hassanali, M., Stothard, P., ..., and Woolsey, J. (Jan, 2006). Drugbank: a comprehensive resource for in silico drug discovery and exploration. *Nucleic Acids Research*, vol. 34, pp. D668–D672.

Yadav, S., Shroff, G., Hassan, E., and Agarwal, P. (2015). Business data fusion. In: *International Conference on Information Fusion*, pp. 1876–1885, Washington, US. IEEE.

Yamada, T., Kato, Y., Maekawa, Y., and Tomiyama, T. (2017). Interactive service for visualizing data association using a self-organizing structure of schemas. In: *International Conference on Service-Oriented Computing and Applications*, pp. 230–233, Kanazawa, JP. IEEE.

Yang, Z., Zheng, B., Li, G., Zhao, X., Zhou, X., and Jensen, C. S. (2020). Adaptive top-k overlap set similarity joins. In: *International Conference on Data Engineering*, pp. 1081–1092, Dallas, US. IEEE.

Zhang, Y. and Ives, Z. G. (Aug, 2019). Juneau: Data lake management for jupyter. *Very Large Databases Endowment*, vol. 12, pp. 1902–1905.

Zhao, Y. (2021). *Metadata Management for Data Lake Governance*. Doctoral Thesis, Université Toulouse, Toulouse, FR.

Zhu, E., Deng, D., Nargesian, F., and Miller, R. J. (2019). Josie: overlap set similarity search for finding joinable tables in data lakes. In: *International Conference on Management of Data*, pp. 847–864, Amsterdam, NL. ACM.

APPENDIX A – CONTAINER CONFIGURATION FOR THE DATA LAKE

To be able to replicate our experiments, it is important to perform some configurations. First we start the containers on our Linux server and check if all containers are correctly initialized:

```
$ docker compose start
$ docker ps
```

The bash must show eight running containers: the hystoryserver, namenode, three datanodes, Neo4J and Jupyter. To configure HDFS to store our dataframes, we need to create a directory and give permission to Apache Nifi to store data in the HDFS directory. To do so, we need to access the running namenode container, create a directory for the datasets, add a user named *nifi* and give permissions to the datasets directory:

```
$ docker exec -it namenode bash
$ hadoop fs -mkdir -p datasets
$ hadoop dfs -put .
$ useradd nifi
$ su hdfs
$ hdfs dfs -mkdir /datasets
$ hdfs dfs -chown nifi:supergroup /datasets
$ hdfs dfs -chmod 755 /datasets
$ hdfs dfs -ls /
$ exit
```

In our local server, we have to select an empty folder to copy, edit and transfer the Hadoop configuration files to the Nifi container, so Nifi can communicate with Hadoop namenode:

```
$ cd /myfolder
$ docker cp namenode:/etc/hadoop/core-site.xml .
$ docker cp namenode:/etc/hadoop/hdfs-site.xml .
$ docker inspect namenode | grep '"IPAddress"' | head -n 1
$ docker inspect namenode | grep '"Hostname"' | head -n 1
```

Next, we edit property fs.default.name or fs.defaultFS (depending on the version of Hadoop) on core-site.xml and set the value we found on hostname or IPAddress. For example: hdfs://66c12aa49c73:9000. Now, we need to copy the configuration files to the Nifi container:

```
$ cd /myfolder
$ docker cp . nifi:/opt/nifi/nifi-current/conf
$ docker cp . nifi:/etc/conf
```

For some dataset such as OMIM and DRUGBANK, we need to manually put it inside Nifi, since they are only available upon request (the ownlers of the dataset do not provide a weblink until now). To do so, we need to access the folder containing the datasets and copy to Nifi container:

```
$ cd /anotherfolder
$ docker cp . nifi:/datasets
```

After that, we can start the Jupyter Notebook webpage, by copying the link:

```
$ docker exec -it jupyter bash
$ jupyter notebook list
```

In the jupyter notebook, we also need to update the IP address or hostname according to namenode.

Pontifícia Universidade Católica do Rio Grande do Sul
Pró-Reitoria de Graduação
Av. Ipiranga, 6681 - Prédio 1 - 3º. andar
Porto Alegre - RS - Brasil
Fone: (51) 3320-3500 - Fax: (51) 3339-1564
E-mail: prograd@pucrs.br
Site: www.pucrs.br