Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

JOONA NEITTAMO

Automated Parsing and Integration of Job Requirements from Public Sources: A Case Study in Excel and SQL

DATA ENGINEERING
2024

ABSTRACT

Author Neittamo Joona	Type of Publication Bachelor's thesis	DATE 4.3.2024
	Number of pages 57	Language of publication: English

Title of publication
Automated Parsing and Integration of Job Requirement from Public Sources: A Case Study in Excel and SQL

Degree Programme
Artificial Intelligence

Abstract

The process of extracting structured information from Excel files is becoming increasingly essential in an era of data-driven decision making. This thesis explores the main challenges associated with extracting and going through data in multiple different patterns changing Excel files and presents versatile strategies to go through these complexities. With the main datasets that hold job requirement information extracted from a web-parser robot, this study addresses the prominent obstacles of location, structure and pattern repetition encountered during the extraction process.

The introductory chapter establishes the context of Excel parsing, emphasizing the importance of efficient data extraction for knowledgeable decision-making. The subsequent literature review provides an in-depth exploration of known tools, methodologies, and impediments pertaining to data extraction from Excel files, sustaining a comprehensive grasp of the subject matter. The chapter centers on location-based challenges investigating the identification of relevant cells and subtle handling of merged cells and spans. Meanwhile, the chapter dedicated to structural challenges talks about the task of normalizing inconsistent data formats and extracting hierarchical data. Lastly, the section focusing on the problems of pattern repeat scrutinizes the discernment of repetitive structures and strategies to effectively manage irregular patterns.

The conclusion chapter joins the findings and implications found throughout the study. The identified challenges and their corresponding solutions collectively contribute to the advancement of data extraction practices, augmenting the efficiency and precision of these processes.

In essence, this thesis furnishes a comprehensive framework for effectively navigating the intricate landscape of extracting information form Excel files.

FOREWORD

This thesis is written from my previous job at Netum, where I was a trainee for the company working in the coding department. This is a written document of the job I finished in summer of 2022 at Netum.

CONTENTS

1 INTRODUCTION 7
1.1 Introduction to the Topic 7
1.2 Background and Motivation 8
1.3 Statement of the Problem 9
1.4 Research Objectives and Questions 10
1.5 Scope and Limitations of the Study 11
2 LITERATURE REVIEW 12
2.1 Overview of Data Extraction in Excel Files 12
2.2 Importance of Information Extraction in Modern Environments 13
2.3 Common Tools and Methods Used for Excel Parsing 15
2.4 Challenges Faced in Extracting Structured Data from Excel Files 17
2.4.1 Location-Based Challenges 17
2.4.1.1 Identifying Relevant Cells 18
2.4.1.2 Handling Merged Cells and Spans 22
2.4.2 Structural Challenges 23
2.4.2.1 Dealing with Inconsistent Formats 24
2.4.2.2 Extracting Hierarchical Data 26
2.4.3 Pattern Repeat Challenges 27
2.4.3.1 Recognizing Repetitive Structures 28
2.4.3.2 Handling Irregular Patterns 31
3 METHODOLOGY 32
3.1 Overview of the Proposed Approach 32
3.2 Data Collection and Preprocessing 32
3.3 Tools and Libraries Used for Parsing Excel Files 33
3.4 Techniques for Identifying Relevant Cells 35
3.5 Approaches for Handling Irregular Patterns 37
4 CASE STUDY AND IMPLEMENTATION 39
4.1 Overview of the Job Requirement Information Dataset 39
4.2 Description of the Excel Files and Their Content 40
4.3 Application of the Proposed Methodology to the Dataset 42
4.4 Results and Insights Obtained from the Implementation 43
4.5 Lessons Learned and Potential Limitations 45
5 CONCLUSION 47
5.1 Summary of the Challenges Discussed in the Thesis 47
5.2 Evaluation of the Proposed Methodology’s Effect 48

5.3 Implications of the Study for Data Extraction Practices 49
5.4 Recommendations for Future Research and Improvement 51
5.5 Conclusion and Final Remarks 53
REFERENCES 56

LIST OF SYMBOLS AND TERMS

Symbol	Description
Parse	extract information from e.g., website / file
AGILE	approach to project management and software development
Pandas	information and data manipulator
DataFrame	digital table for rows and columns (much like excel)
SQL	“tool” / language that can give you the information or create tables depending on context
Cloud service	service that lets you rent a resource such as computing power over the internet which you control yourself
Regex	text patterns for efficient searching, matching and manipulation. Saves time in text tasks.

1 INTRODUCTION

1.1 Introduction to the Topic

This thesis goes through on a comprehensive case study that centers around the automated parsing and seamless integration of job requirements. In the dynamic landscape of data-driven decision-making, the extraction of relevant information from Excel files is a vital challenge. This thesis study talks about the intricate complexities that come during the process of extracting information from Excel-Files, addressing the vital problems of location, structure, and pattern repetition. By looking at the logic behind these challenges and explaining good strategies to mitigate them, this research not only offers insights specific to Excel parsing, but also serves as an essential guide applicable to any other scenario where information would need to be parsed.

In the upcoming chapters, we go through the systematic exploration of these challenges and their solutions. The literature review situates our study withing the bigger context of data extraction, showcasing the importance of overcoming these problems. Subsequent chapters analyse the nuances of location location-based challenges, the intricacies of handling structural irregularities and the methodologies employed in pattern recognition and management. The application of these strategies is shown through a compelling case study, where we showcase the practical implementation of our proposed approaches.

As we go through the complexities withing the domain of data extraction we also highlight the further implications. The knowledge and insights gained from this study not only contribute the enhancements of data extraction practices but also roll across decision support systems and information management strategies.

1.2 Background and Motivation

During a summer job, I was given the essential task involving the extraction of information from multiple Excel files located inside a cloud storage, each meticulously created by a robotic agent. This automated entity actively navigated through an array of websites, thoroughly compiling job applications and their associated requirements. These assortments of valuable data were then deftly organized into Excel files by the robot's intricate process. My role came into play as I embarked on the challenge of parsing these files, extracting their rich content, and orchestrating their transformation into a structured DataFrame. Once this was accomplished, my next step was to seamlessly integrate this information into a self-build SQL table.

"DataFrame is mostly used in data analysis and data manipulation. It lets you store data in tabular form like SQL database, MS Excel, or Google Sheets, making it easier to perform arithmetic operations on the data." (Samdare B, 2024, c. "Different ways to create Pandas Dataframes)

As these challenged came to me, it became clear that the complexities inherent in information extraction will extend well beyond the confines of Excel files alone. It was this realization that made my determination to embark on a comprehensive exploration of these intricacies through this thesis. By unravelling the underlying intricacies and proposing effective solutions, this research strives to not only address the challenged I encountered but also contribute to a bigger understanding of data extraction. Through this journey, I aim to bridge the gap between practical obstacles and effective techniques, ultimately enhancing the landscape of informed decision-making.

1.3 Statement of the Problem

Information extraction is extremely frequent challenge across the spectrum of professional undertakings. Regardless of one's knowledge of coding, Excel proficiency or their first step into the task, grappling intricacies of information extraction will be faced by everyone working on the digital landscape. From seasoned programmers to novices, going through the data flows is abundant with hurdles that span diverse applications. Whether the task involves transforming company emails into a self-designated SQL-table, importing work hours into an application, or meticulously handling Excel files, the fundamental problems persistently manifest in the forms of Location, Pattern repeat, and Structure.

The common nature of these challenges underscores the common need for a comprehensive understanding of their nuances and viable solutions. Inseparably mixed into the fabric of data-driven decision-making, these challenges hinder the seamless acquisition and utilization of valuable information, thereby impending the realization of optimal outcomes. As organizations and individuals alike endeavour to harness the power of data, the resolution of these challenges become paramount.

This thesis goes through the boundaries of job descriptions, roles, and fields. It confronts these fundamental challenges head-on, dissecting their complexities, and elucidating effective methodologies to overcome them. By going through into the intricacies of Location, Pattern repeat, and structure, this study seeks to empower professionals across disciplines, enabling them to navigate the terrain of information extraction with better proficiency.

1.4 Research Objectives and Questions

During my professional journey, the challenges I encountered during data extraction went far beyond the boundaries of Excel files. This thesis looks to go further than those boundaries, offering a comprehensive insight into the intricacies of information extraction from Excel files. The overall goal is to equip readers with a proficient understanding of the nuances involved and empower them to tackle similar challenges across diverse applications.

The main objective of this thesis is to provide a comprehensive understanding of the intricacies underlying information extraction from Excel files. By dissecting the process and explaining effective techniques, this study aims to go through a profound grasp of the fundamental challenges, giving the reader the chance to harness these insights for a myriad data extraction tasks.

Throughout this research, critical questions that encompass the essence of these challenges will be addresses, including:

1. Why do Structure, Pattern repeat, and Location come as the main problems in the landscape of information extraction?
2. How can one accurately identify relevant cells amidst the complexity of data?
3. What strategies can be employed to mitigate the inconsistent patterns within data?
4. How can pattern repetition be recognized, and how should one adapt when confronted with dynamically changing patterns?

These questions are the foundation of our inquiry, guiding the exploration into data extraction difficulties. By working to provide strong answers to these questions, this thesis strives to help the readers with total understanding, positioning them to navigate the challenges of information extraction across different contexts.

1.5 Scope and Limitations of the Study

This chapter outlines the limits that define boundaries of this study, showing meaning on the scope, objectives, and inherent limitations. The limitations are essential to give you the understanding what falls withing the view of this study and to set realistic expectations for the findings and conclusions presented.

Within this study, the chapters give you a comprehensive exploration of key points central to successful data extraction. As you go through the literature review, you will see the main points of this thesis, which revolve around these three key areas:

1. **Location-Based Challenges**: Going through the problems of identifying data points within various structures, giving you the ability to overcome the challenges of data pinpointing.
2. **Structural Challenges**: Making you able to transform inconsistent data formats into organized, usable structures, unlocking the potential buried withing problematic datasets.
3. **Pattern Repeat Challenges**: Guiding you through the intricate landscape of recognizing and managing recurring patterns, a fundamental aspect of efficient data extraction.

The chapters through these topics withing the literature review goes beyond the theory. It encompasses the exploration of practical sub-points encompassing tools, misconceptions, data overview, and optimal approaches. By extracting these insights, you gain proficiency in not only identification and handling but also in the extraction of valuable data.

Although the scope of this study extends to a wide array of extraction scenarios, it’s essential to acknowledge the inherent limitations. While the core principles are universally applicable, efficacy of the presented strategies may vary based on factors such as data complexity, source variability, and the nature of extraction tasks.

2 LITERATURE REVIEW

2.1 Overview of Data Extraction in Excel Files

Excel files stand as a very versatile storage for an array of data, surrounding everything from personal finances and professional records to intricate numerical analyses. Their universal utility is located by their ability to hold data-driven insights and facilitate complex computations within easily navigable spreadsheets. Placed across diverse industries, Excel's excellence extends beyond conventional data storage, serving as a good training ground for artificial intelligence neural networks and a robust platform for wielding intricate equations. Its expansive suite of features enables data entry, formula-based calculations, and customizable visual representations of data making it a stalwart ally in the field of data manipulation.

While manual extraction proves a straightforward option when dealing with solitary files, the landscape becomes more intricate when data is scattered across numerous files. Manual extraction, time-consuming in such instances, necessitates the harnessing of automation tools to streamline the process. Enter APIs such as "pandas" and "xlsx", powerful tools tailored for parsing and interacting with Excel files programmatically. By integrating these APIs, the field of automated data extraction becomes more accessible, probable upon one's proficiency in their usage.

The journey of data extraction, irrespective of the chosen tool or API, commences with a foundational understanding of the basics. In the subsequent chapters, you will go through into the basics that underlie successful data extraction. Our exploration is extended by exploration of Python, known for its versatility, and selection of API's that extend its capabilities. However, the theoretical insights collected are not confined to specific programming languages or tools, they resonate across the spectrum of data extraction scenarios.

“The whole process (also called the data mining process) is almost always iterative. It usually takes many rounds to achieve the final satisfactory result, which is then incorporated into real-world operational tasks.”
(Bing Liu, 2007, p. 6)

This thesis, while going through into the difficulties of data extraction problems and their corresponding solutions, assumes the role of a comprehensive guide. The chapters ahead are sure to unravel the mysteries surrounding data extraction challenges and offer adept strategies for overcoming them. While the narrative unfolds through the figure of python and API usage, the principles and methodologies discussed exceed programming languages, extending their relevance to infinite contexts.

2.2 Importance of Information Extraction in Modern Environments

In the constantly and rapidly evolving view of the modern world, the significance of information extraction goes further than mere convenience; it stands as a very foundation upon which our technological advancement and societal progression link. The job of extracting valuable insights and knowledge from data sources carries a very important role, shaping industries, education, and the preservation of culture in multiple ways. Here are some fuller points:

1. **The Pillar of Technological Advancements**: At the centre of this significance is the crucial role that the information extraction plays in development of artificial intelligence. Think about the world where trained models for AI, the key force in everything from intelligent chatbots to groundbreaking medical diagnostics were held by a lack of effective data. Without the current of structured information to train the neural networks, the AI’s potential to enhance healthcare, streamline automation and make everyday interactions better on websites would not be used. Imagine an app like Excel, the very app reliant on predictive

algorithms to fill all the gaps in data would become worse and the seamless integration of technology into our lives would become worse also.

2. **Empowering Education and Spread Knowledge**: Imagine a universe where education, the principle of progress was held back by a shortage of important data extraction. The large growth of textbooks, digital resources, and educational materials that are the foundation of learning would be flawed by gaps, stalling the growth of ambitious minds. Information extraction stands as a foundation for knowledge spread, making sure that the educational resources are enriched with wideness and depth required to allow generations to come.

3. **Keeper of Cultural Legacies**: Far through the world of technology and education, information extraction serves as a keeper of cultural heritages. The cultural narratives, preserved in recorded forms, always rely on meticulous data extraction to traverse through time. From ancient manuscripts to new artistic expressions, the act of extracting information makes sure that our heritages remain accessible and relatable to upcoming generations.

As you go further into the future, the importance of information extraction will only amplify. Its whole being is the cornerstone of our progression, a link between the past and present.

To put it extremely simply, without information extraction, if you were to view your back account on the internet you would have no money and even simpler, the whole website would not work.

2.3 Common Tools and Methods Used for Excel Parsing

In information extraction, one of the most common tools that people use nowadays is Microsoft Excel itself. In parsing office jobs, the workers don't usually have the skills to use tools such as Python, a tool that has excellent APIs such as Openpyxl, xlrd or pandas, so they are given the task to manipulate cells by hand. Although this may sound horrible, the Formula Bar that is featured in Excel is very efficient in manual work. The Formula Bar is an excellent tool multiple cases in Excel, such as:

1. **Simple Editing**: The Formula Bar is made to be very user-friendly for editing cells and the contents they hold. You can easily access and modify your chosen cell's content by selecting it from the Formula Bar, giving you a safe and quick way to alter your cell data without unintentionally messing up the cell itself.
2. **Multi-Line Editing**: The Formula Bar allows you to edit multiple cells and lines which can make a job shorter.
3. **Auditing Tools**: The Trace Precedents and Trace Dependents buttons in the Formula Auditing group also give you a good way to visualize the relationship between chosen cells and make you understand how formulas are interconnected.
4. **Array Formulas**: The Formula Bar can let you use named ranges in formulas, making the easier to reference certain data in your spreadsheet.

For an experienced data parser, the tools may vary, but there are many more tools available such as:

1. **Pandas**: One of the most common and powerful tools for data manipulation and extraction that is used with Python. Used in data wrangling, cleaning, analysis, and dataframe creation. Pandas is excellent for handling large datasets.

2. **NumPy**: A central package for scientific computing in Python. This Python API offers an adaptable multidimensional array structure, a collection of related entities, and a range of functions for swift array operations encompassing mathematical, logical, and reshaping actions.
3. **Dask**: Library allowing parallel computing used in Python. Used in handling large datasets that don't fit into memory. Excellent for array manipulation with dataframes and structures since it works with Pandas and NumPy.
4. **Openpyxl**: Python API, provides you tools for manipulating spreadsheets with Python code. Allows you to create charts, apply formatting, and perform Excel-tasks automatically with programming.

One of, if not the easiest method to go through cells in Excel automatically is with openpyxl. By first importing the API in Python:

```
import openpyxl
```

Then loading the Excel file with "openpyxl.load_workbook" while assigning it to a variable and making it active:

```
workbook = openpyxl.load_workbook('your_file.xlsx')
worksheet = workbook.active
```

And from there you can make the code iterate through all rows within the file, while giving you a value of the cells:

```
for row in worksheet.iter_rows():
        for cell in row:
                Cell_value = cell.value
```

With this, you can extract information as needed to wherever you want, as the information is now strapped to the "cell_value" variable.

2.4 Challenges Faced in Extracting Structured Data from Excel Files

In the field of information extraction, Excel files are the everywhere as repositories of structured data, holding a massive amount of valuable information. The process of extracting structured data, while often may be straightforward, gives a spectrum of challenges that data practitioners come across in various professional fields.

2.4.1 Location-Based Challenges

In a company workplace, especially for interns or new employees, navigating the problems of accessing data files through many cloud services or a company's SharePoint can present big challenges. These problems are often combined by the limitations of account-side permissions. While granting full access to cloud storage may seem like a solution, this approach introduces its own set of risks, including the increased vulnerability to phishing and fraudulent emails. These risks are not only a concern for individual security but also pose a potential threat to organizations data integrity.

"If these employees do not have a fundamental grounding in the principles of data analytic thinking, they will not really understand what is happening in the business. This lack of understanding is much more damaging in data science projects than in other technical projects" (Foster Provost, 2013, p. 13)

Moreover, reliance on a limited set of API's designed to parse data for specific cloud or SharePoint formats make the already hard challenges even worse. This is particularly true in organizations that customize their data management systems, thereby limiting the compatibility and functionality of available data parsing tools.

Similar challenges are faced in the book Geographic Information Systems and Science (Paul A. Longley, 2015, p. 36)

For independent coders or self-employed professionals, the challenges are even worse. The quality and currency of data might be compromised, access to essential tools and services may be inconsistent, and there may be gaps in technical knowledge. These factors collectively cut off the ability to efficiently parse and utilize data.

Despite these challenges, achieving the capability to consistently parse files through a cloud service or to access high-quality data as an independent worker is a key factor. It necessitates a thorough understanding of the underlying theory and the development of good technical skills. Mastered skills in these areas are essential for accurately identifying and extracting the required data inside cells of specific Excel files.

As you go through addressing the location-based challenges inherent in accessing and parsing data, it becomes imperative to focus on the specifics of cell placement, correct data manipulation and understanding the right way to parse cells. The sub-sequent chapters, “Identifying Relevant Cells” and “Handling Merged Cells and Spans” delve into the nuances of working with complex data structures. These sections aim to equip the reader with the knowledge and techniques necessary for precise data extraction and manipulation, thereby overcoming some of the initial challenges discussed in this chapter.

2.4.1.1 Identifying Relevant Cells

The task of identifying relevant cells in Excel files goes further because of whether these files are intended for data extraction. Instead, it hinges on a variety of factors that influence the complexity and feasibility of the process. Among these, the most significant are:

1. **Structure of the Excel File**: The layout and organization of data within the file play a big role in determining the ease of access to relevant information.

2. **Difficulty of the Given Task**: The difficulties of the objectives for data parsing can greatly affect which cells are considered relevant.
3. **Data Source**: The origin and nature of the data itself influence the relevance of different cells for specific purposes.

These factors are ordered from the most challenging to the most lenient in terms of how they impact the process of identifying relevant cells.

“Data-driven decision-making (DDD) refers to the practice of basing decisions on the analysis of data, rather than purely on intuition.” (Foster Provost, 2013, c. “Introduction to Data-Analytic Thinking”)

The ability to pinpoint the relevant cells in an Excel file requires exceptional problem-solving skills paired with a creative mindset. Look at the structure of the provided Excel sheet example:

ID	Requirement	Mandatory require-ments	Minimum requirement
PV-1	Bachelor’s in engineering	Mandatory	At least a vocational upper secondary qualification
PV-2	English language skills	Mandatory	The expert's proficiency in the Finnish language, assessed according to the Common European Framework of Reference for Languages, is at least level C2.
PV-3	The expert has served as a project manager in software development projects implemented with agile methods.	Mandatory	Experience spanning at least a five-year period, with a total of at least 200 person-days of realized workload.
ID	**Experience and expertise to be scored**	**Mandatory require-ments**	**Basis on scoring**
PP-1	The expert has served in the role of project manager in a digital service development project that has several end-user organizations.	To be scored	1 point per project, maximum 3 points. The scope of the project has been at least 100 person-days. References are considered from the last 36 months counting from the time of submission of the offer.

In the context, it might seem intuitive that the relevant cells are in the second and fourth columns. This thought holds if the task involves manual extraction. However, when designing a parser for such data structures, the task simplifies considerably. The cornerstone of any structures format is the first line, often outlined by the “ID” cell. Surprisingly, this “ID” cell, anchoring the first column, is pivotal. It not only structures the entire dataset but also streamlines the data parsing process for users.

To grasp the concept of identifying the most relevant cells, take a view at the image below:

	A	B	C	D	E	F
1	Table 1					
2	Header 1	Header 2	Header 3	Header 4	Header 5	Header 6
3	1	1.123596	1.123596	19	21.34831	405.618
4	2	2.247191	4.494382	20	22.47191	449.4382
5	3	3.370787	10.11236	21	23.59551	495.5056
6	4	4.494382	17.97753	22	24.7191	543.8202
7	5	5.617978	28.08989	23	25.8427	594.382
8	6	6.741573	40.44944	24	26.96629	647.191
9	7	7.865169	55.05618	25	28.08989	702.2472
10	8	8.988764	71.91011	26	29.21348	759.5506
11	9	10.11236	91.01124	27	30.33708	819.1011
12	10	11.23596	112.3596	28	31.46067	880.8989
13	11	12.35955	135.9551	29	32.58427	944.9438
14	12	13.48315	161.7978	30	33.70787	1011.236
15	13	14.60674	189.8876	31	34.83146	1079.775
16	14	15.73034	220.2247	32	35.95506	1150.562
17	15	16.85393	252.809	33	37.07865	1223.596
18	16	17.97753	287.6404	34	38.20225	1298.876
19	17	19.10112	324.7191	35	39.32584	1376.404
20	18	20.22472	364.0449	36	40.44944	1456.18
21	AVG	10.67416	131.6479	27.5	30.89888	879.9625
22	SD	5.829357	113.9834	5.188127	5.829357	321.7432
23	Min	1.123596	1.123596	19	21.34831	405.618
24	Max	20.22472	364.0449	36	40.44944	1456.18
25						

Figure 1: Messy data structure with possibilities of extraction

Despite the lack of informative headers in this sheet, it is still very possible to identify relevant cells for data extraction. The table, organized under “Header 1” challenges conventional parsing approaches due to its unique quantification by Excel. However, “Header 1” serves as an important marker for cell placement, thereby facilitating data navigation. While it may appear hard, the advent of Regular Expressions (ReGeX) simplifies this way of parsing, transforming the task into a very manageable one.

In essence, the relevant cells are identifiable by their role in quantifying and structuring the data sheet. These cells provide a gateway through which data can be efficiently parsed and analysed.

2.4.1.2 Handling Merged Cells and Spans

Merged cells are a common occurrence in Excel files, often resulting from human error, misprints, or minor data corruption. In the process of developing a parser, encountering merged cells while examining sheets and tables is almost inevitable. At first glance, the complexity introduced by merged cells might seem impossible, tempting one to defer their processing in favour of manual handling at a later stage.

Typically, merged cells combine the content of two or more adjacent cells, making it relatively straightforward to identify which cells have been merged and where the original text may be obscured or altered. It is crucial to approach these merged cells with caution, as they can significantly disrupt the consistency of your parsing code across multiple sheets within an Excel file.

Although a multitude of strategies exists for addressing merged cells, the temptation to bypass these cells manually in scenarios of abundant data might seem justifiable. However, this approach gives a critical examination, especially from the perspective of automating data parsing processes. The essence of automation lies in its ability to simplify and standardize the handling of data, ensuring reliability and efficiency across diverse datasets.

In the context of automation, the decision to manually skip over files with merged cells warrants a nuanced consideration. While manual intervention may offer a temporary workaround for specific cases, it undermines the fundamental objectives of automation: consistency, scalability, and time efficiency. Reliance on manual processes introduces variability and potential for error,

challenging the integrity of the parsed data and the overall efficacy of the parsing tool.

To enhance the strength of automated parsing systems, it becomes imperative to integrate sophisticated strategies capable of intelligently identifying, interpreting, and processing merged cells.

2.4.2 Structural Challenges

Structural challenges in information extraction, whether undertaken individually or within an organization context, invariably relay on the quality of the underlying data. As described in the previous section “2.4.1.1 Identifying Relevant Cells” while certain Excel sheets exhibit commendable structural integrity and extractability (despite the second one being less structured yet manageable), such instances are not mainly guaranteed.

Contrary to the assumption that data within Excel sheets are cleanly organized for future applicability, reality often presents a severe difference. Data, particularly from corporate repositories or public databases, does not always maintain a consistent quality or structure, complicating its utility for future endeavours. The anticipation of data’s relevance and usability is fraught in uncertainties. Even when one encounters datasets with initially clean and well-defines structures, the likelihood of sustaining such orderliness in sequent records remains minimal. This inconsistency mainly stems from automated data collection methods employed by numerous entities. Reliance on robotics and similar technologies, though efficient, introduced a high susceptibility to errors. Consequently, what begins as structured data can rapidly devolve even from the smallest of mistakes.

“Data attributes are retrieved from the content compiled rather than from sources. Unstructured data is prone to errors, inaccuracies, indexing

complications, missing values, and unclear structure.” (Asmitha Arokiamary, 2023, c. Unstructured data)

This inherent unpredictability underscored the necessity for robust methods capable of navigating the predicament of inconsistently formatted data, a subject explored in the next chapter “Dealing with Inconsistent Formats”. Both upcoming chapters 2.4.2.1 & 2.4.2.2 aim to give the reader an understanding of the larger problems and what a good structure is data extraction for data management.

2.4.2.1 Dealing with Inconsistent Formats

It is very important to understand what structures are inconsistent during the extraction of information. The challenges about to be presented by inconsistent formats in Excel underscore the importance of robust data management practices. While the flexibility of Excel is one of its greatest strengths, without proper oversight, it can lead to significant obstacles in information extraction. By acknowledging these challenges and implementing strategic measures to combat them, you can enhance your data analysis capabilities. Inconsistent formats bring many challenges and for example, some of those are:

1. **Data Quality and Reliability**: Varied formats can obscure data quality issues, making it hard to identify and correct errors or inconsistencies withing the data itself.
2. **Integration Issues**: Inconsistent data formats can cause problems when trying to integrate data from multiple Excel files into a single database or analysis tool.
3. **Reportable Data**: For organizations subject to regulatory requirements regarding data management and reporting, inconsistent formats can pose a risk to compliance. Difficulties in extracting accurate and reliable data can lead to reporting errors and potential regulatory penalties.

These are just some of the challenges that you can face when you come across inconsistencies during data extraction. Note that although these challenges can seem like Repeating Patterns, it is not about that. This chapter is about the inconsistency of structures, which focuses on the challenges of the slight errors in the formatting inside sheets, look at this:

NUM-BER	RATING	1	TIER	YEAR	
1	1	1	1	1	MONEY
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	SCORE	1	1	1

Figure 2: Inconsistent structure

Even at the first glance you will notice the inconsistency of the structure. While an inconsistency like this may appear extremely rarely, similar although smaller inconsistencies will appear most of the time. Even from these small inconsistencies, the data can suffer due to the automation of a parser as you may have errors which can preclude further extraction.

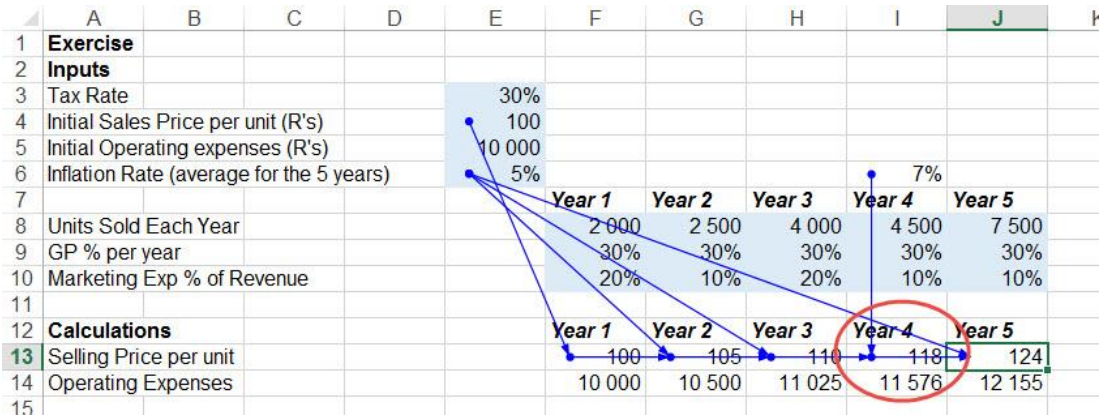


Figure 3: Clean dataset in the context of visual clarity

Extracting information from a sheet built like this is extremely difficult. While you might think that the data on the right is easily extractable since it has a

good foundation by headers, rest of the data on the sheet is too scattered to create a functioning parser as there is no way to know how many different sheets or files you have the parse.

2.4.2.2 Extracting Hierarchical Data

Dealing with formats that appear inconsistent is difficult because even if you manage to parse data that is useful, for future endeavours you need to think if your parser can do the same if data continue to be inconsistent. Most often the inconsistencies come from the relevant data and cells being too scattered on a sheet, which makes you have to loop multiple times through the sheet searching for the relevant headers which can then be used to gather data into a dataframe, also if the data is too inconsistent with the rest of the data of the sheets or even Excel files, it can impair to efficiency of the process rather than just having small amounts of wrong data.

Data bloat, while in extremely small amounts may sound feasible, although it is almost impossible to get rid of during the automated process, can still be very detrimental. You may think that small amounts of data bloat can be ignored, and while you may be right, it all depends on the context where data is used.

Imagine a hospital that has given you inconsistent formats to extract information from, during your extraction you have not found any strategies to deal with, so your 3% within the data is bloated. While it might not sound bad as it is highly possible that you may have an extreme number of megabytes of data, it can have serious consequences:

1. **Misdiagnosis and Incorrect Treatment**: If a patient's medical records contain inaccurate information about their allergies, past medical history, or current medications, healthcare providers might misdiagnose the patient's condition and prescribe incorrect treatments. This could

lead to adverse drug reactions, worsening of the patient's condition, or even fatal outcomes.

2. **Impact on Public Health Decisions and Research**: Medical data is often used in epidemiological studies and public health decisions. Inaccurate data can lead to incorrect conclusions about disease trends, effectiveness of treatments, and public health policies.
3. **Inefficient Use of Resources**: Bad data can lead to unnecessary diagnostic tests and procedures if a patient's history is not accurately recorded. This not only increases healthcare costs but also exposes patients to unnecessary risks associated with certain diagnostic procedures.

Keeping incorrect data within a database is always a bad decision. Removing it is easy, but if you are handling immense amounts of data, it also is more efficient to be automated, but requires more work and time.

Inaccurate data, during data analytics is easily dealt with when using ‘Pandas’. Pandas offers many ways to deal and find bloat, for example. “df.dropna()” or filling missing values with “df.fillna()”

“**Use the attribute mean or median for all samples belonging to the same class as the given tuple**: For example, if classifying customers according to credit risk, we may replace the missing value with the mean income value for customers in the same credit risk category as that of the given tuple. If the data distribution for a given class is skewed, the median value is a better choice.” (Jiawei Han, 2011, p. 88)

2.4.3 Pattern Repeat Challenges

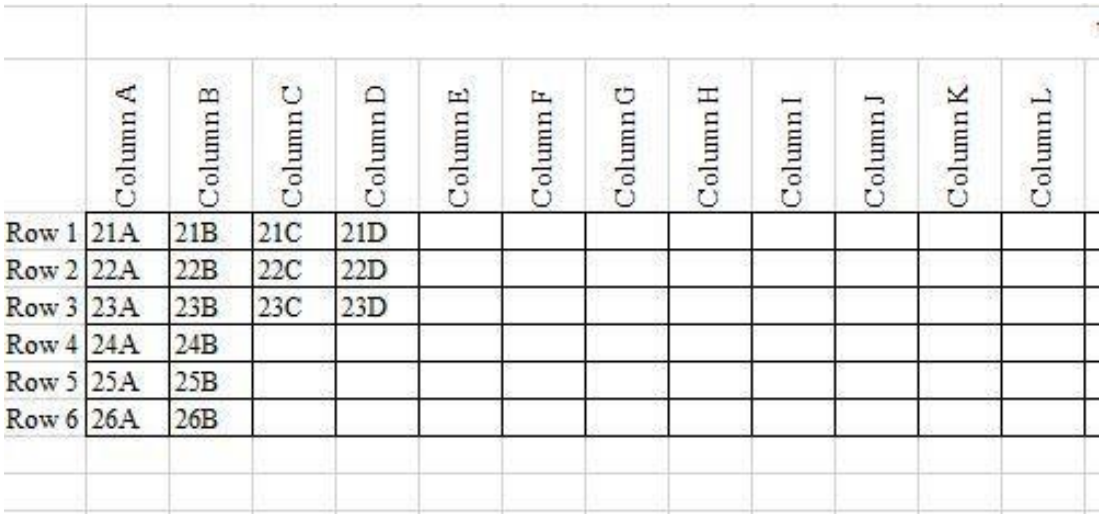
Like the challenges encountered with location-based data, the hurdles associated with repeating patterns are intrinsically linked to the quality of data. When

sourcing data for public websites, researchers often encounter vast quantities of raw, unprocessed information. Typically, this data is not readily extractable due to the preference for publishing large volumes raw data rather than curating and cleaning it for user-friendly access across multiple files. Moreover, the necessity of continuously looping through cloud storage and SharePoint platforms introduces a heightened risk of data loss or even irreversible damage, necessitating elevated access privileges to company files.

Mimicking the issues presented by location-based challenges, even if an independent coder were to discover a substantial amount of clean data, the probability that it’s on a website from which extraction is possible, while possible, is notably low, consider the various factors involved. A further impediment to successful data extraction is the common practice of Excel files containing multiple sheets. These frequently house disparate data structures, which often lack consistent data format. This variability compounds the complexity of extracting usable information, underscoring the intricate nature of navigating repeating pattern challenges in data extraction processes.

2.4.3.1 Recognizing Repetitive Structures

While you may think that regonization of repetitive structures focuses on different sheets and files having the same structure, what is really focuses on is finding the key parts that connect these structures from all the different structures to make data extraction as simple as possible.

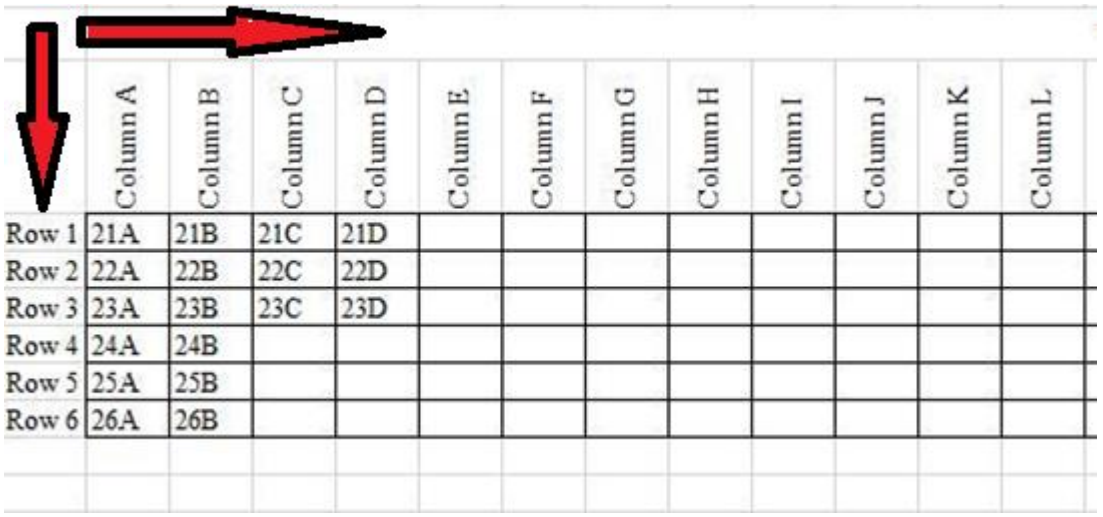


	Column A	Column B	Column C	Column D	Column E	Column F	Column G	Column H	Column I	Column J	Column K	Column L
Row 1	21A	21B	21C	21D								
Row 2	22A	22B	22C	22D								
Row 3	23A	23B	23C	23D								
Row 4	24A	24B										
Row 5	25A	25B										
Row 6	26A	26B										

Figure 4: Continual pattern throughout a structure

The excel structure on the image presented is perfect. Rows use their own numbering and columns use their own coded order. Data in this form is easily extractable, but finding a repeating pattern like this may be difficult. To understand the key to figuring out the simplest way to extract data, you must find a key component within the structures, whether is repeats or not. Even if data is scattered, it is still data, and to build data structures, you, or any automized excel creating robot must use some of the few data structure techniques to make data readable, which in almost 90% of cases this is happens.

The simplest key is often to look for the largest squared data. Here is an example of the key from the previous image:



	Column A	Column B	Column C	Column D	Column E	Column F	Column G	Column H	Column I	Column J	Column K	Column L
Row 1	21A	21B	21C	21D								
Row 2	22A	22B	22C	22D								
Row 3	23A	23B	23C	23D								
Row 4	24A	24B										
Row 5	25A	25B										
Row 6	26A	26B										

Figure 4: Strategy to look for clean extractable structures

This type of formatting is something you have seen multiple times if you have downloaded or looked at any possible excel structure in your life. If a data structure has (most often) an empty cell where the wide part of the and follows the arrows, it should be the most important area of the sheet if it's meant for storing data (excluding cells that count everything together. Also consider the extraction task). Even in the images presented in the chapter "2.4.1.1 Identifying Relevant Cells", both structures follow this exact technique.

Recognizing repetitive structures is very easy as almost at all times Excel sheets offer user created columns or rows that hold data. From these points you can go on with extracting the information, but what if the next sheet is built differently?

A quick way to start looking for a key point in an Excel file that doesn't look repetitive, is to know what type of information it holds. Take for example an Excel file that focuses on storing data about car crashes. The first sheet would probably hold data of the people that were involved in those specific car crashes and the second sheet would hold their cars warranties or insurances. From these two sheets (although more could exist), try to find something that connects them both. If a "Row 1" type key point does not exist for tables in both columns, try to look for data that both sheets have, like a "YEAR".

A similar technique has been applied in "Information Extraction: Techniques and Challenges" (Ralph Grishman, 1997, p. 9)

In short, try to look for a key feature that both or multiple sheets share. If you are scraping data of the same type, the Excel files most often will contain the same key features even if the structure changes.

2.4.3.2 Handling Irregular Patterns

Handling irregular patterns is more manageable than dealing with inconsistent formats. While not having irregular patterns will make extraction easier, with them present it is still very manageable.

A good way to handle irregular patterns is to go through multiple files and see what the median structure is. Most often you will find many different structures if you see even couple of files or sheets, and in these cases if you put together multiple structures side by side and compare their differences, it becomes easy to see the commonality and start creating an extractor for these structures.

Within these files once you have found the average structure type, you can use Regular Expressions, which has been talked about earlier. ReGeX helps in this case as you can skip looping through cells and instead jump to the key point on the sheet like explained in the last chapter. Although Excel does not support ReGeX, it is very possible to create a working way with coding languages like Python.

Using Python opens many different possibilities while creating extraction easier and from these possibilities, one of them is by using the search functions that are built in with Excel. With ‘SEARCH’ and ‘SEARCHB’ you can parse through text and find specific cells containing key words.

3 METHODOLOGY

3.1 Overview of the Proposed Approach

During the project, the proposed approach was to create a data extraction parser with Python to go through multiple folders which all included multiple Excel files holding specific data with multiple sheets. These files were located within a SharePoint with other data, so the code had to be created for a specific location.

The Excel files contained different data structures which had to be built a specific extraction method through key points within the different sheets. After each sheet and file, the data parsed was transformed into string to a ready-made DataFrame with corresponding headers. The DataFrame was then sorted and sent to a SQL Server to store the data with company-built tools.

3.2 Data Collection and Preprocessing

The data collection was made using a Python code. This code specifically targeted a folder inside the company SharePoint, where then it went through multiple folders using the Python Import called ‘os’. To enter this SharePoint using a coding language, a company made API had to imported to authenticate and give access. After gaining access to the SharePoint, the files were then copied inside a self-made folder for safe use to not damage existing content.

The code then went through all folders and processed the data extraction one by one from each Excel file. The API ‘xlrd’ was used to navigate through the Excel sheets. During this process, a keyword list was maintained containing multiple different keywords referencing common words seen in the sheets to know if the sheet had extractable data.

The code went through all cells until it found a cell containing the keyword “ID” or “NRO”. From the first file it opened, it extracted the cells contained right from the “ID” cell and stored them as string for the DataFrame created with the API ‘pandas’, which stored all extracted data for the rest of the collection. The location of the keyword cell was always kept as a variable, so that when the code would arrive at a null cell at the end of the table, it would coordinate back to the keyword cell while going to a lower cell for further extraction.

After the code had gone through all the files, it would grab the role code from each folder which was contained within a text file. This role code would then be imported and linked to the corresponding data inside the DataFrame. From there, the DataFrame was imported row by row to a SQL server using a company made authentication import. Inside the SQL server, the data was cleaned and sorted.

3.3 Tools and Libraries Used for Parsing Excel Files

During the data collection, many tools and Python focused libraries were used. Some of the most used within the project were:

- **Visual Studio Code**: Crucial for its strong code editing and debugging capabilities, Visual Studio Code provided an integrated development environment that streamlines the coding process. With features like syntax highlighting, code completion, and version control, it made the script development and maintenance agile. The extensive marketplace of extensions allowed seamless integration with Python and SQL, making Visual Studio Code a central hub for project’s coding activities.

- **Python**: Standing out for its simplicity and powerful data manipulation capabilities, Python served as a backbone of the project. Its versatility in handling various data formats, combines with an extensive library ecosystem, made Python the ideal programming language for

processing and analysing Excel files. The language's readability and ease of use enabled rapid development and iteration, critical in adapting to the evolving project requirements.

- **Xlrd**: A key library specializing in reading data from Excel files, Xlrd was instrumental in the initial stages of the project. It allowed efficient extraction of worksheets, reading of cell values, and handling of Excel formulas. Xlrd's capability to navigate Excel file structures made it possible to access and extract necessary data without Microsoft Excel, streamlining the data ingestion process.

- **Pandas**: 'Pandas' played a pivotal role with its powerful data structures and analysis tools. After data extraction with Xlrd, Pandas was used for data transformation, cleaning, and analysis. Its DataFrame object facilitated complex data manipulation tasks, such as merging, reshaping, and aggregation, with minimal code. Pandas significantly reduced the preparation time for data analysis, enhancing project efficiency.

- **SQL**: Crucial for storing, querying, and managing the extracted data, SQL was used once the data was cleaned and processed. Its querying capabilities allowed for sophisticated data analysis, enabling insights and report generation based on the extracted Excel data. The ability to interact with SQL databases directly from Python scripts integrated the data workflow seamlessly, making SQL an essential component of the project's data strategy.

"Two commonly used libraries are pandas and openpyxl. Pandas is a powerful data manipulation and analysis library that supports various file formats, including spreadsheets. Openpyxl is specifically designed for working with Excel files and offers features to read, modify, and create spreadsheets." (Shaumik Daityari, 2021, c. "What is the Python library for parsing Excel files?")

3.4 Techniques for Identifying Relevant Cells

As earlier stated, the relevant cell depends entirely on the amount and theme of the data. Let's say you find the relevant cell that for example. is near important data and is using the arrow technique like in the chapter "2.4.3.1 Recognizing Repetitive Structures". To make process easier, you can use the beginner friendly code shown in the chapter "2.3 Common Tools and methods Used for Excel Parsing", and pair it up with using ReGeX. Here is an example of a scenario with unclean data and a way to parse it:

							Monday	points
	U11	Points	Name				Tuesday	11
	Monday	61	Michael				Wednesday	
	Tuesday	812	Jackson				Thursday	
	Wednesday	24	Trevor				Friday	
	Thursday	661	Franklin				Saturday	
	Friday	2252	Michael				Sunday	1
	Saturday	9235	NaN					
	Sunday	1236	Jax					
Monday								
Tuesday		Monday	3		3			
Wednesday		Tuesday			Monday	1	Monday	
Thursday		Wednesday		2	Tuesday		Tuesday	
Friday		Thursday			Wednesday		Wednesday	
Saturday					Thursday		Thursday	
Sunday		3			Friday		Friday	
				11	Saturday	2	Saturday	
					Sunday		Sunday	

Figure 6: Very messy data sheet

As you can see in the presented image, the data you would want is in the top left corner, but what makes it difficult is that there are bits of data that makes the extraction "challenging".

Start by creating a simple code that makes the Excel file readable with 'openpyxl'. Note that you must specify the file-path on your own, and make sure that you have 'openpyxl', 'ReGeX' and 'pandas' installed through PIP3. This can be done in command prompt after downloading PIP:

```
pip3 install openpyxl
pip3 install regex
pip3 install pandas
```

```
from openpyxl import load_workbook
import re
import pandas as pd

your_file_path = 'uncleandatabook1.xlsx'
excelfile = load_workbook(filename=your_file_path)
sheet = excelfile['Sheet1']
```

After this, you should create a section in your code for storing data for the header columns, keeping track of the data, and making a specific pattern with ‘ReGex’ for the relevant cell. If you are unsure with how to create a ‘ReGeX’ for your own specific need, the use of ChatGPT prompting can help you.

```
header_found = False
data = []
headers = []
pattern = r'^[A-Za-z][0-9]{2}$'
```

This ‘ReGeX’ pattern can find a three-character cell which has the first character as a letter and two last characters as numbers. After this, you can use this pattern to go through cells and locate the cell that matches this pattern.

```
compiled_pattern = re.compile(pattern)
for row in sheet.iter_rows():
    for cell in row:
        if compiled_pattern.match(str(cell.value)):
            if not header_found:
                headers = [cell.value]
                header_found = True
            else:
                data.append([cell.value])
            break
    if header_found:
        break
```

With this code, you can effectively find the cell and use it to parse data in your own way. The code given can work for many different scenarios, as long as

the compiled pattern is changed because the theory of the code will still stay the same.

3.5 Approaches for Handling Irregular Patterns

In the exploration of strategies for managing irregular patterns, the methodologies employed are consistent with those applied to pertinent data structures, as illustrated in the concluding segment of the preceding code snippet, which utilizes identical data storage techniques. The forthcoming code example further elucidates this approach:

```
for row in sheet.iter_rows(min_row=1,
max_col=sheet.max_column, values_only=True):
    if header_found:
        data.append(row[col_index:col_index+2])
    for cell in row:
        if cell and re.search(pattern, str(cell)):
            col_index = row.index(cell) + 1
            headers = row[col_index:col_index+2]
            header_found = True
            break

    if header_found and len(headers) > 0:
        header_found = 'data'
```

Initially, the code establishes a loop to traverse the rows across all columns within an Excel document. For each cell encountered, the algorithm assesses whether the cell's content aligns with the predefined pattern defined in the previous chapter. Upon detecting a match, the algorithm successfully identifies the relevant data. Subsequently, it extracts the column headers, which are then aggregated into a Python list for subsequent utilization. Following the accumulation of header names, the script proceeds to parse the data. This parsing continues until all pertinent data beneath the specified headers has been processed, at which point the operation concludes.

```
df_from_excel = pd.DataFrame(data, columns=headers)
df_from_excel.head()
```

Thereafter, the parsed data is organized into a DataFrame using the 'pandas' library, facilitating a structured and comprehensible presentation of the information.

Points	Name
61	Michael
812	Jackson
24	Trevor
661	Franklin
2,252	Michael
9,235	NaN
1,236	Jax

Figure 7: Extracted data

The coding paradigm presented above is adaptable to a variety of data extraction techniques. It remains effective irrespective of whether the crucial data is positioned beneath the dataset or aligned horizontally, necessitating minimal adjustments to accommodate these variations. For individuals lacking proficiency in programming or unfamiliar with Python, the assistance of ChatGPT, an artificial intelligence-powered virtual assistant, can prove invaluable. ChatGPT can modify the code structure given adequate context, making it accessible even to novices. The assistant enhances comprehension by providing detailed instructions on code application and elucidating the purpose of specific code segments through comments, thereby demystifying the coding process.

“Developed by OpenAI, ChatGPT has been trained on a vast corpus of text, absorbing the knowledge and patterns of language from countless sources. It can analyse prompts, understand context, and generate human-like responses.” (Zibtek, 2023, c. “Just How Good is ChatGPT at Writing Code?”)

4 CASE STUDY AND IMPLEMENTATION

4.1 Overview of the Job Requirement Information Dataset

The dataset under consideration presents a structured compilation of job requirements for various positions, primarily focusing on project management roles within the IT sector. The dataset is organized into a tabular format within an Excel spreadsheet, making it conducive for parsing and automated analysis.

Each entry in the dataset is identified by a unique ID and is accompanied by a detailed description of the job requirement, which is categorized into several attributes. These attributes include:

- **Requirement**: This column outlines the specific skill or experience needed for the job role. For example, it lists requirements such as "English language skills" and "Experience in software development projects."

- **Mandatory Requirements**: This column clarifies whether the requirement is mandatory, which signifies its critical importance for the job role.

- **Minimum Requirement**: Describes the basic level of qualification or experience needed to be considered for the role, such as "at least a vocational upper secondary qualification" or "proficiency in the Finnish language."

- **Response to Minimum Requirement**: This field is meant for applicants to fill in, demonstrating how they meet the minimum requirement.

- **Description of Requirement Satisfaction**: Here, applicants are expected to detail how they fulfill the requirement, providing evidence or explanations where necessary.

- **Basis on Scoring**: This is applicable for requirements where expertise is scored, offering a metric for evaluation like "1 point per project, maximum 3 points."

- **Response to Scorable Requirement**: This section is reserved for applicant input, where they justify the points claimed based on their experience and expertise.

The dataset is designed to facilitate the systematic evaluation of job applications by providing clear and quantifiable criteria for essential and desirable qualifications. It is intended for use by hiring managers or HR departments to assess candidates' fit for project management roles in technology-driven environments.

The subsequent chapters of the thesis will delve into the specifics of the Excel file structure, the application of the methodology for parsing and integrating this dataset, the results derived from this implementation, and an evaluation of the lessons learned, and any potential limitations encountered in the process.

4.2 Description of the Excel Files and Their Content

The Excel file under analysis was generated through an automated process designed to parse multiple job application websites. A sophisticated robot, or web scraper, was programmed to methodically visit specified job listing pages, identify and extract key job requirement information, and subsequently compile this data into a structured format suitable for analysis. This automated approach ensures a systematic and unbiased collection of data, significantly reducing the manual effort and time typically associated with data entry.

The dataset for the study is encapsulated in an Excel workbook that is meticulously organized into several sheets, each corresponding to a specific role

within the IT project lifecycle. This arrangement not only simplifies the parsing of data but also aligns with the intuitive understanding of roles within the industry. Below is a brief overview of the content within each sheet:

1. **Instructions**: This sheet contains guidelines on how to interpret and use the data across the workbook. It may include conventions, definitions, and any assumptions made within the context of the dataset.
2. **Project Manager**: The data under this category would typically outline the job requirements and qualifications for a project manager role. This includes mandatory requirements, scoring basis for experience, and other relevant criteria for evaluating the expertise of a candidate.
3. **Architect**: In this sheet, the focus shifts to the architectural aspect of IT projects. It probably details the necessary skills, experience, and the evaluation metrics used to assess an architect's proficiency in designing system structures.
4. **Application Developer**: This section delves into the qualifications for application developers, highlighting programming languages, development frameworks, and other technical competencies required for the role.
5. **Requirements Expert**: The focus here would be on the abilities to define project requirements, including communication with stakeholders, elicitation of requirements, and documentation skills.
6. **UI/UX Expert**: Lastly, this sheet covers the criteria for evaluating the expertise of UI/UX professionals, focusing on design principles, user experience considerations, and related technological skills.

Each sheet in the workbook is formatted to facilitate the automated parsing of information, with consistent use of columns for requirement IDs, descriptions, mandatory and scoring stipulations, and responses. This structured approach not only streamlines the process of data extraction but also aids in maintaining uniformity across different job roles.

The content within these sheets provides a comprehensive baseline for analysing job requirements across a variety of IT roles. It forms the foundation

upon which automated parsing and integration methods can be applied and tested, allowing for a case study that has practical implications in the realm of talent acquisition and human resource management within the technology sector.

4.3 Application of the Proposed Methodology to the Dataset

In applying the proposed methodology to our dataset, we used a combination of Python scripting and SQL database management to parse and integrate job requirements data from Excel files. This intricate process began with the automated extraction of data using Python. The script specifically targeted structured content across various sheets representing different IT roles, from 'Instructions' to 'UI/UX Expert.' Each sheet was meticulously parsed to ensure the comprehensive extraction of relevant data points, such as requirement IDs, descriptions, and mandatory qualifications.

The Python script used libraries like pandas for reading Excel files and manipulating data frames, and SQLAlchemy for database interactions. This facilitated a seamless transition from raw data extraction to structured data processing. For example, using pandas, the script read each sheet into a data frame, filtered columns based on predefined criteria, and cleaned the data to remove any inconsistencies. This preprocessing was crucial for standardizing the format of our data for analysis.

Subsequently, the cleaned and structured data was integrated into a SQL database, leveraging SQLAlchemy to create a database schema that mirrored the essential structure of our dataset. This step involved defining tables for different IT roles, with fields corresponding to requirement IDs, descriptions, and qualifications. SQL commands were then executed to insert the processed data into these tables, ensuring that our database accurately represented the extracted information.

This application of Python and SQL demonstrated the effectiveness of our methodology in handling diverse data structures, showcasing its potential to streamline data extraction tasks in human resource management within the technology sector. By automating the parsing and integration of complex datasets, our approach not only validated its utility but also highlighted its adaptability and efficiency. The use of Python for data manipulation and SQL for data storage provided a robust foundation for further analysis, enabling us to draw meaningful insights from the aggregated job requirements data.

4.4 Results and Insights Obtained from the Implementation

This chapter delves into the empirical findings derived from the application of the proposed methodology for automated parsing and integration of job requirements from public sources. The primary focus is on the extraction of data from Excel files, highlighting the intricacies and challenges encountered, as well as the efficacy of the strategies employed to address these challenges.

The implementation phase underscored a fundamental challenge in the realm of data extraction: the inherent complexity of parsing information from Excel files, or indeed, any file format. Despite these obstacles, the methodologies outlined in previous chapters facilitated a streamlined approach to overcoming these hurdles. Three pivotal aspects emerged as crucial to the successful extraction of data: pattern recognition, relevancy determination, and precise location identification within the files.

The task of extracting data from Excel files was found to be nuanced and challenging, aligning with the anticipated difficulties. The versatility of Python, alongside other programming languages, played a critical role in navigating these challenges. The ability to manipulate Excel files extensively allows for the overcoming of potential obstacles, provided there is sufficient expertise in utilizing these programming tools.

A comparative analysis revealed a stark contrast between the proposed method and conventional data extraction techniques, which typically rely on the clean, structured nature of data and the use of Excel formulas for manipulation. While Excel formulas are adept at handling straightforward data manipulations, they lack the comprehensive capabilities offered by programming languages like Python, especially in the context of complex data extraction scenarios.
One of the more surprising outcomes was the relative rarity of encountering unclean data, contradicting the common expectation within the field. Furthermore, the process of parsing merged cells, a task anticipated to be complex, was notably simplified using Python. The most significant challenge encountered was not the data extraction itself but accessing the locations where the Excel files were stored, highlighting a logistical rather than technical hurdle.

The implementation benefited immensely from the inherent structure of the Excel files, particularly the cells' identification markers ranging from "PV-1 to 6" to "NRO." These identifiers significantly streamlined the data extraction process, enabling the automated scripts to efficiently locate and extract the necessary information.

The findings from this study illuminate the complexities and challenges of data extraction from Excel files, underscoring the efficacy of the proposed methodologies in addressing these challenges. The versatility of programming languages like Python has proven to be a cornerstone in overcoming the limitations of traditional data extraction methods, offering a more flexible and powerful approach to parsing, and integrating job requirements from public sources.

4.5 Lessons Learned and Potential Limitations

The utilization of the XLRD API alongside Python's Pandas and NumPy libraries for parsing Excel files and handling data manipulation proved to be highly effective. The ability to identify relevant cells and systematically parse information into a structured DataFrame facilitated a seamless transition from raw data to a format ready for SQL integration. This process underscores the utility of Python libraries in simplifying data handling tasks, making it a recommended approach for similar future projects.

A notable discovery was the capability to address and rectify errors in merged Excel files through code. This finding is significant as it offers a methodological approach to overcome what is often perceived as a manual and error-prone task, thus enhancing the efficiency of data preprocessing, and ensuring the integrity of the dataset.

A notable discovery was the capability to address and rectify errors in merged Excel files through code. This finding is significant as it offers a methodological approach to overcome what is often perceived as a manual and error-prone task, thus enhancing the efficiency of data preprocessing, and ensuring the integrity of the dataset.

One of the primary challenges encountered was the initial lack of understanding regarding which cells contained pertinent information. This challenge highlights a limitation in the methodology's applicability to datasets where the structure is not well-defined or consistently maintained across documents. Future applications of this methodology may require initial manual review or the development of more sophisticated detection algorithms to identify relevant data points effectively.

The project's approach assumed that all data would possess a certain level of structure necessary for parsing. This assumption, while generally met, may not hold true for all datasets, particularly those that are highly unstructured or inconsistently formatted. Such scenarios present a limitation to the current

methodology's effectiveness and may necessitate additional preprocessing steps or alternative parsing strategies.

The methodology's reliance on specific tools and APIs, while beneficial for this project, may limit its applicability to a broader range of data sources or formats not supported by these tools. Future efforts may need to explore more versatile or adaptable parsing solutions to accommodate a wider variety of data types and sources.

The journey of parsing and integrating job requirement information from public sources has been both enlightening and challenging. The lessons learned from this experience provide valuable insights into the capabilities and limitations of automated data processing methodologies. While the current approach has demonstrated success within its applied context, ongoing refinement and adaptation are necessary to address the highlighted limitations and to enhance its applicability to more diverse and complex data environments. As the field of data parsing and integration continues to evolve, so too will the methodologies and tools at our disposal, promising ever-greater efficiency and adaptability in the face of burgeoning data challenges.

5 CONCLUSION

5.1 Summary of the Challenges Discussed in the Thesis

This thesis delves into the multifaceted challenges encountered in data extraction, parsing, and management, emphasizing the hurdles faced by both organizations and independent professionals in the digital era. Central to our discussion are three primary categories of challenges: location-based challenges, structural challenges, and pattern repeat challenges, each presenting unique obstacles to efficient data handling and utilization.

Location-Based Challenges underscore the intricacies of navigating cloud services and SharePoint, particularly for new employees or interns within a corporate setting. The restrictions imposed by account-side permissions, coupled with the risks associated with granting full access to cloud storage, such as increased susceptibility to phishing and fraudulent activities, highlight the delicate balance between accessibility and security. The reliance on a limited set of APIs further exacerbates these challenges, especially in environments with customized data management systems. For independent coders or self-employed professionals, these issues are magnified by inconsistent access to tools, gaps in technical knowledge, and the compromised quality of data.

Structural Challenges revolve around the inherent quality and organization of data. Contrary to the assumption of clean, well-structured Excel sheets, the reality often reveals a stark contrast with data lacking consistent quality or structure. This inconsistency, largely stemming from automated data collection methods, poses significant hurdles to maintaining orderliness in data records. The unpredictability of data formats necessitates robust methods for dealing with inconsistently formatted data, a subject further explored in subsequent chapters.

Pattern Repeat Challenges are intricately linked to the quality of data and the complexities of extracting information from repeating patterns. The preference

for publishing large volumes of raw data on public websites complicates the extraction process due to the lack of curation. Additionally, the risk of data loss and the variability in Excel file structures further impede the extraction of usable information.

5.2 Evaluation of the Proposed Methodology’s Effect

The adopted methodology, combining Python scripting and SQL database management, was designed to parse and integrate job requirements data from Excel files. Its effectiveness in achieving research objectives was systematically evaluated.

The Python script, leveraging libraries like pandas and SQLAlchemy, efficiently extracted and processed structured content from various IT role sheets. This ensured comprehensive extraction of key data points, laying the foundation for subsequent analyses. Pandas facilitated the seamless transition from raw data extraction to structured data frames. The script filtered columns based on predefined criteria, cleaning the data to eliminate inconsistencies. This preprocessing standardized data format for robust analysis.

The cleaned data was integrated into a SQL database using SQLAlchemy, creating a schema mirroring the essential dataset structure. SQL commands executed the insertion of processed data, ensuring database accuracy. The methodology's impact was evident in its ability to systematically extract, process, and integrate diverse job requirements data. It automated processes, ensuring data consistency and standardized formatting, proving efficient for subsequent analyses.

Acknowledging limitations and challenges, the methodology's evaluation provides a balanced perspective on its strengths and areas for potential improvement. The next section explores the broader implications of this methodology

for data extraction practices, shedding light on its significance and broader impact.

5.3 Implications of the Study for Data Extraction Practices

The comprehensive case study undertaken in this thesis delves deep into the complexities of automated parsing and integration of job requirements, particularly focusing on the challenges posed by data extraction from Excel files. The identified issues related to location, structure, and pattern repetition open a gateway for understanding and addressing broader challenges in data-driven decision-making scenarios. By unravelling the logic behind these challenges and proposing effective strategies, this research not only provides insights specific to Excel parsing but also serves as a foundational guide applicable to diverse scenarios where information extraction is a necessity.

Originating from a practical experience involving the extraction of information from cloud-stored Excel files created by a robotic agent, the study extends beyond the confines of this specific scenario. Recognizing the inherent complexities in information extraction, the research aims to contribute not only to overcoming immediate challenges but also to fostering a deeper understanding of data extraction practices. The journey outlined in this thesis bridges the gap between practical obstacles and effective techniques, ultimately enhancing the landscape of informed decision-making.

The significance of information extraction in modern environments transcends the confines of Excel files alone. As the study unfolds, it becomes evident that the challenges and solutions explored have broader implications across diverse data-driven applications. The methodologies developed are not limited to a specific platform but offer a universal approach to information extraction challenges in various contexts.

At the core of the study's importance lies its contribution to technological advancements, particularly in the development of artificial intelligence. The structured information extracted plays a pivotal role in training neural networks, powering AI applications that enhance healthcare, streamline automation, and improve everyday interactions. The study's findings have far-reaching effects on the evolution of predictive algorithms and technologies integral to our daily lives.

Information extraction emerges as a fundamental pillar supporting education and the dissemination of knowledge. In a hypothetical world without effective data extraction, the growth of textbooks, digital resources, and educational materials would be hindered, affecting the development of ambitious minds. The study underscores the role of information extraction in empowering education and ensuring the enrichment of educational resources with the depth required for future generations.

Beyond technology and education, the study recognizes information extraction as a keeper of cultural heritages. From ancient manuscripts to contemporary artistic expressions, meticulous data extraction ensures the preservation and accessibility of cultural narratives. The study emphasizes the role of information extraction in safeguarding our cultural legacies for generations to come.

As we progress into the future, the importance of information extraction is set to amplify. It stands as the cornerstone of our technological and societal progression, serving as a link between the past and present. Simply put, without information extraction, essential online functions, from financial transactions to website functionality, would be compromised.

5.4 Recommendations for Future Research and Improvement

The exploration into automated parsing and integration of job requirements from public sources has unearthed several crucial insights with far-reaching implications for the broader landscape of data extraction practices. Understanding these insights is essential for navigating the complexities of data extraction in various contexts.

A fundamental aspect brought to light is the significance of computing power in data extraction endeavors. Depending on the nature of the data being parsed, decisions must be made regarding the adequacy of computing resources. This consideration extends beyond the mere parsing process, influencing the efficiency and scalability of data extraction strategies.

The study illuminated a persistent dilemma in data extraction—balancing the need for precision with the desire for versatility. Practitioners must weigh the benefits of a code tailored for a specific structure against the potential trade-offs when dealing with diverse data structures. This dynamic decision-making process underscores the inherent challenges in data extraction.

The findings of this study resonate with current industry practices in data extraction, particularly in the differentiation between database-oriented extraction and the challenges posed by data formatting programs like Excel or PDFs.

Common tools like Integrate or Stitch dominate the landscape of database-oriented data extraction, providing efficient solutions for structured data. However, the study underscores that for unstructured formats such as Excel or PDFs, relying solely on in-app tools or manual extraction may limit functionality, hindering repeatability and scalability.

The study's strategies offer an alternative perspective for data extraction from commonly used applications. Recommendations from earlier chapters emphasize coding strategies over manual processes, providing a more robust and

repeatable approach to extracting structured data from seemingly challenging formats.

Building on the study's insights, several recommendations for improving data extraction practices emerge, emphasizing the importance of continuous learning and skill development.

The foremost recommendation centers on enhancing coding skills. As the study demonstrates, a higher proficiency in coding significantly contributes to the effectiveness of data extraction processes. Continuous learning and upskilling in programming languages can empower practitioners to tackle a broader range of data extraction challenges.

Beyond coding, a deepening understanding of data structures and formatting nuances proves pivotal. Continuous knowledge expansion in the domain of data extraction ensures practitioners can adapt their strategies to varying data landscapes.

The methods and strategies devised in this study extend beyond the realm of job requirements, finding applicability in diverse domains.
The study's approach can be seamlessly applied to multiple domains, providing a flexible framework for parsing, and integrating data from varied sources. While the specific challenges may differ, the strategies outlined can be adapted and modified to suit the unique characteristics of different datasets.

Regardless of the domain, the challenges identified—such as understanding relevant cells and adapting to unstructured data—remain universal. This universality underscores the importance of developing adaptable strategies capable of addressing common hurdles in diverse data extraction scenarios.

The information gleaned from this study introduces a fresh perspective on decision-making in the context of data extraction, challenging conventional recommendations.

Unlike prevalent advice favoring ready-made apps for data extraction, this study advocates coding as a robust alternative. While various platforms may suggest user-friendly apps, they often fall short when faced with more complex data structures. The study's insights encourage decision-makers to consider coding solutions, thereby fostering a more nuanced and effective approach to data extraction.

In conclusion, the implications of this study extend far beyond the realm of automated parsing of job requirements. They provide a roadmap for practitioners and decision-makers to navigate the challenges of data extraction in a rapidly evolving technological landscape. By prioritizing coding proficiency, adapting strategies, and recognizing the universality of certain challenges, professionals can enhance their data extraction practices and contribute to a more efficient and resilient data ecosystem.

5.5 Conclusion and Final Remarks

In the ever-evolving landscape of data extraction, this study embarked on a journey to unravel the intricacies associated with parsing job requirements from public sources. The exploration illuminated the inherent difficulties in data extraction, particularly the nuanced challenges posed by the repetitive structures prevalent in various datasets. However, with a foundation in coding languages and a keen awareness of key challenges, the seemingly formidable task of data extraction becomes navigable.

The overarching reflection from this endeavour is that while data extraction can be daunting, armed with a profound understanding of coding languages and the ability to confront specific challenges head-on, it transforms into a manageable and even empowering process. Beyond the scope of job requirement parsing, the challenges discussed in this study echo through various tasks that necessitate data extraction, each presenting its unique set of obstacles. The

lessons learned here serve as a valuable guide for anyone navigating the intricate terrain of data processing.

The primary achievement of this study lies in providing a comprehensive understanding of the key problems associated with data extraction and, more importantly, equipping individuals with the knowledge to address these challenges effectively. By dissecting the intricacies of parsing job requirements from public sources, this research contributes a nuanced perspective to the broader field of data processing.

The study's insights into the balance between precision and versatility, the significance of computing power, and the value of coding proficiency collectively form a valuable arsenal for professionals engaging in data extraction. The contributions made here extend beyond the immediate context of job requirements, offering a blueprint for tackling data extraction challenges across diverse domains and applications.

As technology continues its relentless march forward, the landscape of data analysis and software evolves in tandem. While the methodologies presented in this study provide a robust foundation, future research should direct its focus towards harnessing the advancing possibilities in data extraction.

The key problems identified in this study will persist, irrespective of technological advancements. Therefore, future directions should explore innovative approaches to enhance the capabilities of data extraction, adapting to the changing formats and structures adopted by companies to manage their data. Research endeavours could delve into optimizing existing methodologies and developing new strategies to extract valuable insights from increasingly complex datasets.

The completion of this thesis would not have been possible without the support and collaboration of my esteemed colleagues. Their trust in assigning the data extraction task, coupled with their valuable insights and assistance during the creation of the code, played an integral role in shaping this research. It is with

sincere gratitude that I acknowledge the collaborative efforts of my colleagues, as their contributions significantly enriched the depth and breadth of this study.

In conclusion, this journey through the challenges and triumphs of data extraction underscores the resilience and adaptability required in the face of evolving data landscapes. As we move forward, armed with the lessons learned and insights gained, the path to effective data extraction becomes not just a challenge, but an opportunity for continuous growth and improvement.

REFERENCES

Samdare B. (2024) Different ways to create Pandas Dataframe. Geeksfor-Geeks. https://www.geeksforgeeks.org/different-ways-to-create-pandas-data-frame/

Bing Liu. (2007). What is Data Mining? M.J. Carey, & S. Ceri (Eds.), Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data (pp.6). Springer. https://sirius.cs.put.poznan.pl/~inf89721/Seminarium/Web_Data_Mining__2nd_Edition__Exploring_Hyperlinks__Contents__and_Usage_Data.pdf

Foster Provost & Tom Fawcett. (2013). Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking. Mike Loukides & Meghan Blanchette (Eds.), Data-Analytic Thinking (pp.13). O'Reilly Media. https://github.com/tohweizhong/pdf-dump/blob/master/01%20Books/data%20science%20for%20business.pdf

Paul A. Longley. (2017). Geographic Information Systems and Science. Jorge Rocha & Patrícia Abrantes (Eds.), Location Analytics and Modelling (pp.36). IntechOpen. https://repositorio.ul.pt/bitstream/10451/43193/1/Rocha_Abrantes_2019.pdf

Asmitha Arokiamary. (2023). What are the challenges of data extraction, and how to overcome them? Xtract. https://xtract.io/blog/data-extraction-challenges-and-how-to-overcome-them/

Jiawei Han, Micheline Kamber, & Jian Pei. (2011). Data Mining: Concepts and Techniques. Jiawei Han (Ed.), Data Cleaning (Third Edition, pp. 88). https://myweb.sabanciuniv.edu/rdehkharghani/files/2016/02/The-Morgan-Kaufmann-Series-in-Data-Management-Systems-Jiawei-Han-Micheline-Kamber-Jian-Pei-Data-Mining.-Concepts-and-Techniques-3rd-Edition-Morgan-Kaufmann-2011.pdf

Ralph Grishman. (1997). Information Extraction: Techniques and Challenges. Ralph Grishman (Ed.), Basic Techniques (pp. 3). Springer Verlag https://ccc.inaoep.mx/~villasen/bib/GrishmanInformationExtraction.pdf

Shaumik Daityari. (2021). Using Python to Parse Spreadsheet Data. Sitepoint. https://www.sitepoint.com/using-python-parse-spreadsheet-data/

Zibtek. (2023). Just How Good Is ChatGPT at Writing Code? LinkedIn. https://www.linkedin.com/pulse/just-how-good-chatgpt-writing-code-zibtek/