VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Komal Azram Raja

# DOMAIN SPECIFIC DATA QUALITY FRAMEWORK

School of Technology
2024

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Cloud Based Software Engineering

## ABSTRACT

| | |
|---|---|
| Author | Komal Azram Raja |
| Title | Domain Specific Data Quality Framework |
| Year | 2024 |
| Language | English |
| Pages | 37+1 Appendix |
| Name of Supervisor | Johan Dams |

Organizations are generating large volumes of data, including structured and unstructured types from various sources. Financial institutions, which rely heavily on accurate data, face unique challenges that require more knowledge related to the domain. This thesis develops a framework using a Retrieval-Augmented Generation (RAG) model, trained on financial data quality issues and solutions, to provide a context-aware data validation approach. The research aims to bridge the gap between generic data quality checks and context aware approach. By comparing the RAG model's performance with traditional data quality management tools, this study evaluates its effectiveness based on metrics like timeliness, consistency, and completeness. The findings suggest that integrating domain-specific knowledge into data validation processes can provide relevant information and can be utilized in financial data quality management. Future work will focus on improving and implementing the framework’s scalability and performance through using more relevant data and real-world testing.

Keywords Data Quality, LLM, Domain-Driven, Financial data quality

# CONTENTS

**LIST OF TABLES**



**LIST OF FIGURES**

**LIST OF APPENDICES**

# 1 INTRODUCTION

In the last decade, we have seen the data-driven buzzword becoming the reality of organizations. The amount of data that's getting generated is increasing at a much faster rate. According to a forecast made by Statista, the amount of data generated, consumed, and stored is going to be more than 180 zettabytes. The data is not only limited to traditional databases and data warehouses anymore; we have IoT devices, sensors, and various other data collection methods to collect both structured and unstructured data. (Petroc Taylor, 2023). However, the trust in the quality of data is still questionable and organizations spend time and resources to ensure the datasets meet the quality requirements. The poor-quality data leads to missed opportunities and wrong decisions. (Kober et al., 2011). The life cycle of a data pipeline consists of data validation checks that ensure that data meets the quality requirements. Every data point tells a story and is unique. While designing a data validation pipeline the context or data domain should be taken into consideration. Every domain has specific limitations that can't be compromised. If the domain-specific requirements are taken into consideration the data validation process can provide additional context for the problem and can leverage the historical data. This approach can help to design a better data-cleaning strategy.

## 1.1 Background Information

Financial institutions are dependent on enormous amounts of data that come from a variety of sources, such as market analytics, transactions, client contacts, and regulatory reports. Ensuring the integrity of this data is crucial since low-quality data can result in inaccurate analysis, non-compliance with regulations, monetary losses, and reputational damage. Poor quality data caused Equifax, a publicly traded credit reporting agency, to send lenders inaccurate credit scores on millions of customers. (Equifax Statement on Recent Coding Issue, 2022). According to Gartner every year, poor data quality costs organizations an average

of $12.9 million (Manasi Sakpal, 2021). However, because of their generic approach and lack of contextual awareness, typical data quality technologies frequently fail to meet the unique needs of the financial domain. The data quality solutions currently in use are based on generic algorithms to find common problems with data quality, like outliers, duplicates, and missing values. These technologies are helpful for fundamental data quality checks, but they are not sophisticated enough to handle the contextual needs and domain-specific issues that are present in financial data. Without knowing the details of a quarterly financial closure activity, a generic algorithm can, for instance, label a high-value transaction as an anomaly, resulting in false positives and inefficiencies.

A Retrieval-Augmented Generation (RAG) system trained specifically on data quality problems and solutions faced by the organization overtime can address these needs effectively. Unlike general-purpose tools, a RAG system leverages a comprehensive knowledge base tailored to the specific challenges faced in dealing with finance data. This helps the system give better recommendations and solutions, making data validation more accurate. (Divatia, 2023)

Table 1 shows the solutions that various data quality tools offer under different scenarios, highlighting their effectiveness and contextual relevance.

| **Tool** | **Feature** | **Generic Algorithm** | **Areas for Improvement in Financial Domain** | **Example** |
|---|---|---|---|---|
| Talend Data Quality | Duplicate Detection | Exact or duplicate matching of fields<br>This method relies on comparing fields to identify exact matches or duplicates.<br>(*Unique/duplicates* Talend Studio User Guide) | Doesn't provide context-specific guidance on approach usage | May miss detecting transactions that involve minor variations such as slight differences in transaction descriptions, timestamps, or amounts |
| Informatica Data Quality | Missing Values Detection | Checks for null or empty values in fields<br>(*Informatica data quality 10.4.0*) | Does not differentiate between critical and non-critical missing data | Missing timestamps in trading data can significantly impact financial analysis and<br>demographic information, such as a customer's secondary email address, may be less critical and not impact the core financial analysis. |

| | | | | |
|---|---|---|---|---|
| Microsoft SQL Server (DQS) | Consistency Checks | Simple cross-field validation rules (*Temporal table system consistency checks - SQL server*) | Can be improved for complex relationships across financial datasets | Ensuring consistency in diverse datasets that consider exchange rates, market conditions, and regulatory requirements. |

**Table 1:** Comparison of Data Quality Tools

### 1.2 Scope

This thesis aims to develop a framework to provide context during data validation by providing domain-specific knowledge for the specific use-case. The framework will bridge the gap between data quality tools and domain-knowledge. This will enhance the data cleaning and data validation processes during data quality management. Specifically, the research will focus on the financial industry and will utilize a Retrieval-Augmented Generation (RAG) model.

The main objective of the thesis is to train the RAG model on a dataset comprising various data quality issues encountered in financial data, along with their corresponding solutions. The trained model will then be applied to new data cleaning and validation tasks within the financial domain to see if it provides any context during the data validation process.

To test the effectiveness of the proposed solution, RAGAS framework will be used to access the quality of response. The performance of the RAG model will be compared with existing data quality tools. The comparison will be based on specific data quality metrics such as timeliness, consistency, and completeness.

This comparison will help determine whether the context-aware approach provides any useful recommendations during data validation.

# 2 LITERATURE REVIEW

In this thesis we will explore data quality and see if there is any relation between domain and quality of data. There are various methodologies to assess the quality of data like statistical analysis, data monitoring, data profiling, etc. each tailored for specific use cases. Adding context to a scenario can make the classification of data quality dimensions disputed, as different dimensions are prioritized based on the context. The six data quality dimensions (completeness, accuracy, uniqueness, consistency, timeliness, and validity) are often prioritized differently depending on the context. (Batini, Cappiello, Francalanci, & Maurino, 2009)

When dealing with financial data, accuracy and timeliness are critical dimensions, often focusing on real-time accuracy. Data lineage also plays an important role in tracking the origin and relationship of data in the finance domain. Techniques to improve data quality can be data-driven or process-driven.The data-driven approach updates data with the most recent values, suitable for short-term goals but not ideal for long-term planning. On the otherhand process-driven approach focuses on changing the data collection process or storage format. The domain and type of data being used can help identify the correct dimensions, improve quality, and reduce costs. (Batini, Cappiello, Francalanci, & Maurino, 2009)

In recent decades, the discussion around data quality has significantly increased in both academic research and real-world applications. The importance of data quality has become more evident with the spread of data-driven approaches and the development of advanced business intelligence tools like Tableau and Power BI, as well as market intelligence platforms like NetBase Quid and Crunchbase Pro (Santhosh Kumar, 2023).

The shift towards fit-for-use data quality emphasizes the importance of user viewpoints and domain-specific requirements. Reliable data quality measures should satisfy normalization, adaptivity, scalability, interpretability, and cardinality requirements. These measures are practical for evaluating and

enhancing data quality across various domains, particularly in AI, where model effectiveness depends on data quality (Santhosh Kumar, 2023).

Data quality is crucial for decision-making, innovation, and business success. However, it poses significant challenges due to the proliferation of data sources and the velocity of data generation. Key challenges include data volume and velocity, data silos, integration issues, governance, security and privacy, and human error (Pansara, 2023).

Ensuring data quality at scale is an important task. Organizations must address these challenges to ensure the integrity, reliability, and utility of the information that fuels their decisions and actions. Research aims to understand data quality dimensions, challenges, impacts on decision-making, assessment tools, emerging trends, and case studies (Pansara, 2023).

Domain-specific solutions provide a better approach for data-driven products. BI maturity models define how well organizations use BI for data-driven models, but current models do not consider the details of clinical and financial data. Research on domain-specific BI maturity models, particularly in healthcare, provides a six-step framework to enhance traditional models, aligning with the idea of using a domain-specific data quality framework (Bharadwaj et al., 2017).

In AI, domain-specific knowledge bases (KB) are more accurate than generic open-domain KBs. However, incorporating domain knowledge, designing efficient workflows, monitoring and maintaining data quality, and querying the knowledge bases are major challenges. A content service system can address these challenges, enhancing AI-driven applications' capabilities (Bertossi & Geerts, 2020a).

Data quality is measured by interacting with databases using queries and scripts. The quality of data used in training impacts AI and ML models' outcomes. Explainable AI suggests using domain-specific knowledge to build applications,

improving data quality and building reliable and trustworthy AI models (Bertossi & Geerts, 2020b).

# 3 RESEARCH DESIGN

In this chapter, we outline the research design employed to for enhancing data quality validation in the financial sector. This includes the rationale behind adopting a RAG-based solution, the specific research objectives, and the methodologies for data collection and evaluation.

## 3.1 Rationale for a RAG-based Solution

This thesis presents a Retrieval-Augmented Generation (RAG)-based strategy that uses a large language model (LLM) trained on data quality concerns and their solutions relevant to the financial sector to provide context during the data validation process.

Compared to general-purpose models such as ChatGPT, this technique offers distinct advantages, especially in the finance industry where privacy, control, and customization are essential. Privacy is a critical concern in finance due to the sensitive nature of financial data, which includes personal and transaction information. Control over data processing and storage ensures that financial institutions can comply with internal policies and regulatory requirements. Customization allows the model to be fine-tuned to address specific financial queries and scenarios, enhancing its relevance and effectiveness (Takyar, 2023).

A RAG-based strategy is particularly effective in ensuring regulatory compliance. Some financial regulations may not be adequately satisfied by public LLMs like ChatGPT, which may not support on-premises data processing or adherence to particular jurisdictional standards. Regulations such as the General Data Protection Regulation (GDPR) in Europe and the Gramm-Leach-Bliley Act (GLBA) in the United States impose strict data handling and privacy requirements. The option to adopt a local RAG solution enables financial institutions to meet these regulations by processing and storing data within their own controlled environments. (Divatia, 2023).

Data security is one of the main issues facing the finance sector. A local RAG-based model, trained on a company's historical data, offers a more secure and customizable solution compared to cloud-based models. In March of 2023, due to a bug in OpenAI's ChatGPT, a few users were able to view other users' chat titles, underscoring the potential risks associated with cloud-based systems (OpenAI, 2023).

### 3.2 Research Objectives

This research aims to examine the impact of providing context in traditional data quality tools in the financial sector by using a context-aware approach that leverages a Retrieval-Augmented Generation (RAG) model.

1) Construct a well-structured repository of financial data quality issues and their resolutions, sourced through a web-parser and organized for efficient use by the RAG model.

2) Test the theoretical foundations of RAG-based models in enhancing domain-specific data quality solutions for the finance industry.

### 3.3 Data Collection

An essential aspect of this research is data collection, which ensures that the knowledge gathered is comprehensive and relevant for building a strong basis for a domain-specific data quality solution in the financial sector. The objective of this step is to compile detailed information regarding problems with data quality in the financial sector and how they have been addressed.

The General Data Protection Regulations prevents organizations to share their data so collecting real-world dataset for this thesis isn’t possible. In order to build a foundation, we will acquire data from a variety of online sites, making sure that all of the data is appropriately sourced and publicly available. The data sources include:

**Online Articles**: We'll compile business articles from reliable financial websites. These articles will provide case studies and real-world instances of problems with data quality in the finance industry, along with solutions.
**Personal Blogs:** We will source educational materials from the personal blogs of people who have worked with finance data.

To guarantee the transparency of the research, an extensive list of all sources will be included in the appendix. This approach will ensure comprehensive, ethical, and trustworthy research, providing a basis for testing the foundation for a domain-specific data quality process tailored to the financial sector.

## 3.4 Tools and Technologies

To implement a robust data validation framework tailored for the financial sector, several advanced tools and technologies are employed. The following subsections detail the technologies used, their functionalities, and their relevance to the research objectives.

### 3.4.1 Web Scraping

Web scraping techniques facilitate the extraction of large amounts of relevant information from a variety of online sources, including websites, industry publications, reports, and journals. The technique extracts structured data from web pages programmatically. Technologies like Scrapy, BeautifulSoup, and Selenium are used to sift through HTML text and locate relevant content. By developing custom scripts for each data source, web scraping ensures the efficient and methodical collection of data pertinent to the study objectives. After that, preprocessing is done on the data to increase its suitability for research in the future. This entails trimming out extraneous text, removing HTML tags, and organizing the data so that it may be processed further. Web scraping has shown to be a successful technique for obtaining large-scale, timely datasets.

### 3.4.2 Large Language Models and RAG

Large language models (LLMs) are trained on extensive datasets to handle a wide range of applications. These models fall under the larger class of fundamental models within generative AI. They excel at tasks such as writing code snippets, answering questions, summarizing texts, and producing content. LLMs utilize various factors to understand and reproduce language details, providing replies that resemble human communication. They are generally accessible through platforms like Chat-GPT, OpenAI, and Meta.

These open-source LLMs, despite being trained on vast amounts of data, occasionally struggle with context awareness, which can hinder their ability to accurately respond to specialized queries. Additionally, because the data used to train them isn't always up-to-date, their responses may sometimes lack relevance and transparency.

Retrieval-Augmented Generation (RAG) is a technique that enhances the performance of LLMs by incorporating external, reliable knowledge bases into their response process before generating replies. This method leverages the billions of parameters in the extensive data that LLMs are trained on to generate new content for various tasks, including question answering, language translation, and sentence completion.

Over time, LLMs have evolved from basic conversation flows to generating verifiable and tailored responses. By incorporating the latest documents or policies, RAG reduces the need for frequent updates and retraining of models with new data. IBM currently uses RAG to ensure that its internal chatbots for customer service are based on accurate and verified data (IBM Research. (n.d.).

### 3.4.3 Hugging Face

Hugging Face has developed into an essential hub for natural language processing (NLP) enthusiasts and practitioners, offering an extensive range of tools, models,

and resources that promote the field's growth. The Hugging Face ecosystem includes a broad range of pre-trained language models, from small-scale architectures for fine-tuning on specific tasks to large-scale transformer models trained on massive amounts of text data. Numerous NLP applications, such as text production, sentiment analysis, machine translation, and question answering, are built on top of these models. (Hugging Face Hub Documentation).

As the field of NLP continues to expand, Hugging Face is committed to democratizing access to cutting-edge technologies. Their vast open-source ecosystem, which offers a variety of pre-trained models, tools, and resources to both researchers and developers, is a clear indication of this commitment (Hugging Face Hub Documentation).

In this thesis, Hugging Face embeddings will be used to create sentence embeddings. The models from Hugging Face are effective for generating high-quality sentence embeddings, which are essential for the implementation of a Retrieval-Augmented Generation (RAG) solution.

### 3.4.4 Llama 2 by Meta AI

The introduction of Llama 2 by Meta AI is a significant advancement in chat functionality and transparent communication. By providing an "open foundation," this creative paradigm differs from the traditional closed-source approach. This openness translates to readily available architecture data and training sets, allowing researchers and developers to investigate the model's internal workings and tailor it to their needs. (Roumeliotis et al., 2023).

A key concept in Llama 2's philosophy is fine-tuning. While the core model speaks English fluently, fine-tuning allows targeted training on specific activities, including picking up on the nuances of chat conversations. Through this process, Llama 2 becomes more perceptive of the complexities of conversation, enabling it to

generate insightful and relevant comments related to the subject matter. (Roumeliotis et al., 2023)

When building chatbots, security remains a primary consideration. Recognizing this, Meta AI has implemented several safeguards to ensure that Llama 2 operates in a safe and advantageous environment. These techniques most likely involve limiting biases in the model's training set, avoiding the production of offensive content, and ensuring that the responses are truthful and objective. (Roumeliotis et al., 2023)

Llama 2 is currently considered the greatest open-source chat model, outperforming previous models in terms of helpfulness, safety, and overall performance on language understanding tests. The focus on safety and effectiveness makes it a reliable tool for developers aiming to build advanced and secure applications. (Roumeliotis et al., 2023)

### 3.5 Evaluation Plan

The primary objective of this evaluation plan is to assess the effectiveness of the proposed RAG-based (Retrieval-Augmented Generation) framework in managing data quality within the financial industry. This will be achieved by comparing the framework's responses to common data quality queries against the data validation options available in existing data quality tools.

To evaluate the final results, we will use the following test cases sourced from articles written by industry professional and educational blogs on dealing with specific data quality issues in finance data:

**Testcase 1:** This testcase is formulated from Atlan’s guide on dealing with stale data. The testcase is designed to measure the timeliness dimension. The usefulness of the recommendations depends on the relevance and actionable insights provided by the framework. (Atlan, n.d.)

**Testcase 2**: This testcase is formulated from FinanceTrain solution on handling the missing data. The testcase is designed to measure the completeness dimension. The results will be evaluated by the fact that it adds additional context to use strategies for addressing missing customer income data and compare them with standard methods like mean imputation, regression imputation, and multiple imputation. (Finance Train, n.d.)

**Testcase 3:** This testcase is formulated from Blazent guide on dealing with currencies and consistency in finance data. The testcase is designed to measure the consistency dimension. The results will be checked if the end results give any recommendations to validate consistency and compare the validation results with those from current data quality tools. (Blazent, 2016)

The articles that are used to designed these testcases aren't part of training data. To verify this please refer to the data sources provided in the appendix A. The expected output is extracted from these articles and is compared with RAGs output to see if the generated answers are relevant using ragas framework. RAGAS is a framework designed to evaluate the quality of responses generated by RAG models. It includes various metrics to assess the faithfulness, relevance, and contextual accuracy of the generated answers. The following metrics will be used to evaluate the answers:

**Faithfulness:** Measures how accurately the generated answer reflects the retrieved context.

**Answer Relevancy:** Assesses the relevance of the generated answer to the posed question.

**Context Precision:** Evaluates the precision of the context used to generate the answer.

**Context Recall:** Measures the recall of relevant context used in the answer.

**Harmfulness:** Critiques the potential harmfulness of the generated answer.

(Doe, Smith, & Turing, 2023)

# 4 IMPLEMENTATIONS

## 4.1 Project Architecture

The architecture of the proposed RAG-based solution is designed to leverage a large language model (LLM) and an internal knowledge base to provide context-aware responses for data validation tasks in the financial sector. The following diagram illustrates the flow of data and the interaction between different components of the system:

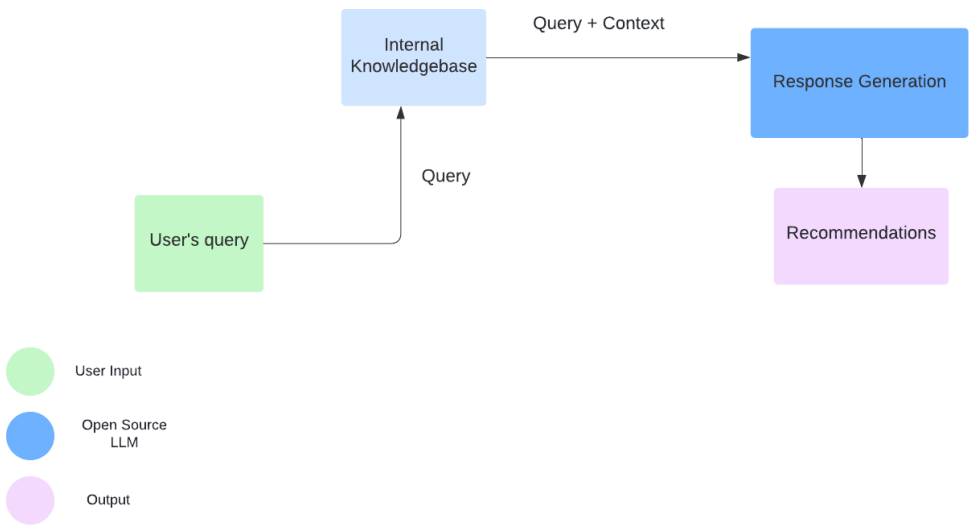


**Figure 1:** Architectural Diagram

## 4.2 Data collection

The data in this project refers to the content related to data quality challenges and their solutions. A web-parser was developed in order to facilitate the process of collecting this data faster. When the parser takes content from websites, it makes sure that only relevant data is recorded—avoiding the ads and other unnecessary details. The parser's primary task is to extract HTML material that has been marked with particular class identifiers that indicate the body of the article.

In order to make the HTML material easier to read and analyze further, it had to be parsed and turned into text. The text data is then stored in the PDF format. Around 12 articles were parsed, offering diverse viewpoints and contemporary examples of data quality issues and their resolutions.

```
def fetch_html_content(url):
    response = requests.get(url)
    return response.text

def parse_article(html_content):

    soup = BeautifulSoup(html_content, "html.parser")
    article_text = ""
    article_content = soup.find("div", class_="entry-content")
    if article_content:
        paragraphs = article_content.find_all("p")
        for paragraph in paragraphs:
            article_text += paragraph.get_text() + "\n\n"
    return article_text
```

**Figure 2:** Code Snippet for Web Scrapper

### 4.3 Knowledgebase and LLMs

Once the content is available it is used to build a knowledgebase for the LLM model. Before diving into the technical process, it's essential to understand two key concepts: embeddings and vector representations.

**Embeddings:** In the context of natural language processing (NLP), embeddings are a way to represent words, phrases, or entire documents as numerical vectors. These vectors capture the semantic meaning of the text, meaning that words with similar meanings will have similar vector representations.

**Vector Representations:** This refers to the transformation of text data into a series of numbers (vectors). These vectors can be thought of as points in a multi-dimensional space. For example, the word "finance" might be represented by a

vector in a 768-dimensional space where each dimension captures some aspect of its meaning.

The data, which is in the form of PDF files, needs to be read and processed. We use SimpleDirectoryReader(), a data connector provided by Llama Index, to read the files from the directory and load them into the code.

```
documents=SimpleDirectoryReader("/content/data").load_data()
```

**Figure 3:** Code Snippet for Loading Knowledgebase

Next embeddings for the text in the documents are created. Embeddings help the LLM understand the context and meaning of the text. A pre-trained sentence transformer from the Hugging Face library called all-mpnet-base-v2. This model maps the text into a 768-dimensional space. Essentially, it converts the text into a vector with 768 numbers, each capturing some aspect of the text's meaning. These embeddings are then used to provide additional context to the LLM. When a user queries the model, it uses these embeddings to understand the context better and provide more accurate and relevant responses.

```
embeddings=LangchainEmbedding(
    HuggingFaceEmbeddings(model_name="sentence-transformers/all-mpnet-base-v2"))
```

**Figure 4:** Code Snippet for Creating Embeddings

To guide the response of the LLM system prompts and query wrappers are defined.

The system_prompt sets the context for the LLM by defining its role and behavior. In this case, the model is being instructed to act as a Q&A assistant. System_prompt variable holds the instructions that are given to the LLM before it starts processing any queries.

The query_wrapper_prompt formats the user's query before it is sent to the LLM for processing. PromptTemplate is a class or function (depending on the

implementation) that helps in creating a template for the queries. It ensures that the queries are formatted in a way that the LLM can understand and process effectively.

The "{query_str}"is a placeholder that indicates where the user's query will be inserted. When the user inputs a query, it replaces {query_str} with the actual query text.

```
system_prompt="""
You are a Q&A assistant. Your are going to answer questions as
accurately as possible based on the instructions and context provided.
"""
## Default format supportable by LLama2
query_wrapper_prompt=PromptTemplate("<|USER|>{query_str}<|ASSISTANT|>")
```

**Figure 5:** Code Snippet for Prompt Template

Let's initializes a large language model (LLM) from Hugging Face with specific configurations. Llama 2 model from the Hugging Face library will be used. The Llama 2 model we use is trained on seven billion parameters. Parameters are parts of the model that are learned from the training data and are used to make predictions. Following parameters are used to initialize this model:

**context_window:** This parameter sets the context window size for the model. It specifies the maximum number of tokens the model can consider in its input context. A larger context window allows the model to take more information into account when generating responses.

**max_new_tokens:** This parameter limits the maximum number of new tokens the model can generate in a single response. It prevents the model from generating excessively long outputs.

**generate_kwargs:** These are keyword arguments passed to the generation function:

**temperature:** Sets the temperature for sampling. A lower temperature (close to 0) makes the output more deterministic and focused, while a higher temperature makes it more random.

**do_sample:** False ensures that the model generates text deterministically rather than sampling from the probability distribution, providing consistent outputs for the same input.

**system_prompt:** This is a variable holding a prompt that sets the initial context or instructions for the model. It guides the model on how to behave or what kind of responses to generate.

**query_wrapper_prompt:** This is another prompt variable that wraps around user queries, likely to format or modify the input query in a specific way before it's processed by the model.

**tokenizer_name:** This specifies the name of the tokenizer to be used. A tokenizer is responsible for converting text into tokens that the model can process. Here, it's using the tokenizer from the "meta-llama/Llama-2-13b-chat-hf" model.

**model_name:** This specifies the name of the model to be used. The model "meta-llama/Llama-2-7b-chat-hf" is a specific variant of the Llama-2 model designed for chat applications.

**device_map:** This parameter automatically maps the model to available hardware devices (e.g., CPU, GPU) for efficient computation. It helps in utilizing the hardware resources optimally.

**model_kwargs**: These are additional keyword arguments for model configuration:

**torch_dtype:** Specifies the data type to be used by the model (16-bit floating-point). This can speed up computation and reduce memory usage.

**load_in_8bit:** True indicates that the model should be loaded in 8-bit precision, further reducing memory usage while maintaining performance.

```
llm = HuggingFaceLLM(
    context_window=4096,
    max_new_tokens=256,
    generate_kwargs={"temperature": 0.0, "do_sample": False},
    system_prompt=system_prompt,
    query_wrapper_prompt=query_wrapper_prompt,
    tokenizer_name="meta-llama/Llama-2-7b-chat-hf",
    model_name="meta-llama/Llama-2-7b-chat-hf",
    device_map="auto",
    # uncomment this if using CUDA to reduce memory usage
    model_kwargs={"torch_dtype": torch.float16 , "load_in_8bit":True}
)
```

**Figure 6:** Code Snippet for Initializing LLM

The service context is initialized to bundle the large language model (LLM) and the embedding model within the Llama Index pipeline. This setup specifies parameters such as chunk_size, which determines the size of text chunks for efficient processing. The index pipeline refers to the sequence of processes that transform and manage the text data, making it ready for quick retrieval and analysis. By doing this, the system ensures efficient data processing and enhances the performance of the LLM in responding to user queries.

```
service_context=ServiceContext.from_defaults(
    chunk_size=1024,
    llm=llm,
    embed_model=embeddings
)
```

**Figure 7:** Code Snippet for Initializing Service Context

The documents are indexed using the VectorStoreIndex to create vector representations of the text. This indexing process transforms the text into a format that can be efficiently searched. The query_engine is then created from the index

using the as_query_engine() method. This query engine is responsible for handling user queries. It uses the indexed vectors to find relevant information and provides context to the large language model (LLM) to generate accurate answers for user queries.

```
index=VectorStoreIndex.from_documents(documents,service_context=service_context)
query_engine=index.as_query_engine()
```

**Figure 8:** Code Snippet for Initializing Service Context

The next steps involve using this query engine to interact with users, answering their questions by retrieving and processing the most relevant information from the model.

### 4.4 User Interaction and Testing

Let's discuss how the users interact with the framework and how it processes their queries to generate queries. This interaction helps gather insights into the data and identify potential problems that need to be addressed. The purpose of the tests is to examine whether the RAG-based solution provides additional context to the problem.

#### 4.4.1 Testcase 1: Timeliness check

**User Query:** How to check if the stock data is not too old and it's not stale?

**Purpose:** Validate the framework's ability to assess data timeliness.

```
response=query_engine.query("how to check if the stocks data is not too old and it's not stale")

[15] print(response)

To check if the stocks data is not too old and not stale, you can use a combination of data cleaning techniques
```

To check if the stocks data is not too old and not stale, you can use a combination of data cleaning techniques and statistical methods. Here are some steps you can follow: 1. Remove duplicates: Remove any duplicate data entries to prevent inconsistencies and errors in your analysis. 2. Fill in missing values: Fill in any missing values in your dataset to ensure that your analysis is based on complete and accurate data. 3. Correct inaccuracies: Correct any inaccuracies in your data to ensure that your insights are reliable and trustworthy. 4. Outlier removal: Identify and remove any outliers in your dataset to prevent distorted insights. 5. Data freshness analysis: Use statistical methods such as the z-score or interquartile range (IQR) to determine if your data is too old or stale. If the data is stale, you can either remove it or transform it using techniques such as winsorization or log transformation. By following these steps, you can ensure that your stocks data is accurate, complete, and fresh, and that your analysis and forecasting are reliable and trustworthy.

**Figure 9:** Testcase 1

### 4.4.2 Test-case 2: Completeness check

**User Query:** A significant portion of customer income data is missing. What are the strategies to deal with this.

**Purpose:** Evaluate the framework's strategies for handling missing data.

```
[19] response=query_engine.query("A significant portion of our customer income data is missing. What are some strat

[20] print(response)

Missing data can be a challenge, but there are ways to address it:

1. Identify the cause of missing data. Is it random (e.g., system errors), systematic (e.g., specific income br
2. Depending on the cause and amount of missing data, you can:
        * Impute missing values using statistical methods like mean/median imputation for random missingness or
        * Remove rows/columns with excessive missing data if it doesn't significantly impact the analysis.
        * Document the missing data for transparency and potential future use.
3. Time-Series Data Challenges:
        * Acknowledge the time-series nature of the data.
        * Use appropriate time series analysis techniques like ARIMA models or machine learning algorithms spec
        * Consider incorporating time-related features into the analysis, such as lags or trends, to capture th
```

**Figure 10:** Testcase 2

### 4.4.3 Test-case 3: Consistency check

**User Query:** The currency column has these ($, €, &&,₱) values. are these valid currencies?

**Purpose:** Test the framework's ability to validate data consistency.

```
[21] response=query_engine.query("The currency column has these ($,€,&&, Φ) values. are these valid currencies")

     Show hidden output

[22] print(response)

     The values you provided ($$, €, &&, Φ) are not valid currencies in the context of finance and accounting. These
     To identify valid currencies in your dataset, you can use a currency identifier list provided by the ISO or oth
     Once you have identified the valid currencies in your dataset, you can convert the data into a standardized for
```

**Figure 11:** Testcase 3

## 4.5 Analysis of results

To evaluate the responses of the proposed RAG-based solution for data quality management in the financial industry, we conducted the tests as mentioned in the section 2.5. The responses generated by the framework is evaluated using RAGA and following results are obtained:

```
eval_questions = [
    "How to check if the stock data is not too old and it's not stale?",
    "A significant portion of customer income data is missing. What are the strategies to deal with this?",
    "The currency column has these ($, €, &&,Φ) values. are these valid currencies?",

]

eval_answers = [
    "To ensure stock data is not too old and stale, compare the timestamps of the last update with the current time and verify that updates increment
    "When a significant portion of customer income data is missing, you can address this issue using several strategies. First, you can use interpola
    " The currency column with values ($, €, &&, Φ) contains both valid and invalid currency symbols. The dollar sign ($) and the euro sign (€) are v
]
```

**Figure 12:** Code Snippet for Expected Response

```
eval_answers = [[a] for a in eval_answers]

from ragas.metrics import (
    faithfulness,
    answer_relevancy,
    context_precision,
    context_recall,
)
from ragas.metrics.critique import harmfulness
metrics = [
    faithfulness,
    answer_relevancy,
    context_precision,
    context_recall,
    harmfulness,
]
result = await evaluate(query_engine, metrics, eval_questions, eval_answers)
print(result)
```

```
{faithfulness': 0.7000, 'answer_relevancy': 0.9550, 'context_precision': 0.2335, 'context_recall': 0.9800, 'harmfulness': 0.0
000}
```

**Figure 13:** Code Snippet for Evaluation

The results demonstrate that while the RAG model performs exceptionally well in terms of answer relevancy and context recall, there are areas for improvement in faithfulness and context precision. Enhancing these aspects could lead to more accurate and precise answers. The absence of harmful content is a positive sign, reflecting the reliability of the model in generating safe responses. Future iterations of the model should focus on refining the retrieval process to improve context precision and ensure even higher fidelity in the generated answers.

Table 2 compares the framework's responses to the data quality checks provided by existing data quality tools, examining whether it provides any contextual information.

| Query | Functionality Provided by Data Quality Tool | Framework's Response |
|---|---|---|
| Check if stock data is stale | Basic timestamp check (Freshness: DBT developer hub 2024) | Provides a step-by-step explanation to approach the problem |
| Handle missing income data | General imputation methods (Informatica, n.d., "Replace Missing Values" section) | An explanation of what can go wrong and what measures to take |
| Validate currency symbols | Simple regex validation (Temporal table system consistency checks - SQL server) | Provides the context by mentioning the ISO resource to check for cross-validation |

**Table 2:** Analysis of Results

The study suggests that by offering context-aware solutions specifically designed for the financial sector, the RAG-based framework has the potential to enhance data quality control. Existing solutions usually use timestamp checks for the query on determining whether stock data is outdated, which can miss important contextual elements. The RAG-based framework, on the other hand, suggests to take a more comprehensive approach by giving a thorough, stepwise explanation that takes market behavior into account.

The framework not only gives the techniques to remove the missing data but also suggests further analysis to investigate the cause. For validating currency symbols, existing tools generally use simple regex validation, which may fail to identify invalid symbols accurately. The framework's context-aware validation approach suggests to use the currency standards defined by ISO to correctly filter the

incorrect symbols. However, it fails to give a detailed answer that shows which ones are correct symbols and which ones you should look for.

# 5 LIMITATIONS

## 5.1 Enhancement of Domain-Specific Knowledge

The thesis established a foundation that incorporates domain-specific knowledge into data quality tools focusing on the financial industry. However, the current implementation is limited to the publicly available finance data quality issues and it can be improved by training the knowledge base with more data quality issues specific to finance data, the model can be improved.

## 5.2 Scalability and Performance Constraints

The use of Google Colab limits scalability and performance due to memory constraints and the need for GPU resources. This environment is not optimal for building a production ready tool or performing high-performance computations necessary for robust data quality tools.

# 6 FUTURE ENHANCEMENTS

## 6.1 Training on expanded knowledge base

To enhance this solution, it is necessary to train it with a large amount of data related to data quality issues encountered in the finance domain during data validation. The blueprint for this dataset already exists in the form of Jira logs and Notion entries. These logs and entries contain detailed records of data quality issues, resolutions, and the context in which they occurred, providing a comprehensive foundation for creating the training dataset. Real-world testing with financial datasets will be essential to validate the framework's effectiveness and robustness in a production setting. Training on real-world data will ensure that the framework can handle real-time data quality issues faced by financial organizations, thereby improving its reliability and performance.

## 6.2 Benefits for the Organizations

Organizations can significantly benefit from integrating the enhanced RAG model with their existing data quality tools or using it as a standalone solution. By training local RAG models on historical data quality issues, organizations can develop context-aware, domain-specific data validation processes. This approach not only helps in data management but also ensures data security by processing sensitive information locally. The flexibility of deploying RAG models either integrated with current systems or as standalone solutions allows organizations to tailor the implementation to their specific operational needs, leading to better understanding and resolution of data quality problems.

# 7 CONCLUSIONS

This thesis explores the potential of a Retrieval-Augmented Generation (RAG) model to provide context during data quality assurance in the financial industry. By training the RAG model on a dataset of financial data quality issues and solutions, the study examines whether this approach is useful in the data validation process.

The primary contribution of this research is the development of a RAG-based model's ability to offer contextual insights during data validation. The study highlights the limitations of generic data quality tools and explains how a domain-specific approach could help address financial data's unique challenges.

This research provides a preliminary approach of using a RAG-based model for context-aware data quality validation in finance. While the current implementation is limited by the Google Colab environment, future work will focus on enhancing performance and scalability through local RAG-based platforms and comprehensive real-world testing. The findings suggest that incorporating a RAG based model can provide domain-specific knowledge into data validation processes.

## REFERENCES

Ataccama. (n.d.). Get started with data quality. Ataccama ONE. https://docs.ataccama.com/one/latest/getting-started/get-started-with-data-quality.html

Atlan. (n.d.). How to measure stale data: Here are 8 simple steps. Atlan. https://atlan.com/stale-data/#how-is-stale-data-measured-here-are-8-simple-steps

Batini, C., Cappiello, C., Francalanci, C., & Maurino, A. (2009). Methodologies for data quality assessment and improvement. ACM Computing Surveys, 41(3), 1–52. https://doi.org/10.1145/1541880.1541883

Bertossi, L., & Geerts, F. (2020a). Data quality and explainable AI. Journal of Data and Information Quality, 12(2), 1–9. https://doi.org/10.1145/3386687

Bertossi, L., & Geerts, F. (2020b). Data quality and explainable AI. Journal of Data and Information Quality, 12(2), 1–9. https://doi.org/10.1145/3386687

Blazent. (2016, December 1). The dimensions of data quality: Currency and consistency. Blazent. https://blazent.com/dimensions-data-quality-currency-consistency/

Creation and interaction with large-scale domain-specific knowledge bases. Proceedings of the VLDB Endowment, 10(12), 1965–1968. https://doi.org/10.14778/3137765.3137820

dbt Developer Hub. (2024, May 23). Freshness. https://docs.getdbt.com/reference/resource-properties/freshness

Divatia, A. (2023, December 22). How financial services companies secure data in a RAG GenAI environment. Finextra. https://www.finextra.com/blogposting/25439/how-financial-services-companies-secure-data-in-a-rag-genai-environment

Doe, J., Smith, J., & Turing, A. (2023). RAGAS: Automated Evaluation of Retrieval Augmented Generation. arXiv. https://arxiv.org/abs/2309.15217

Equifax. (n.d.). Equifax statement on recent coding issue. https://www.equifax.com/newsroom/all-news/-/story/equifax-statement-on-recent-coding-issue/

Equifax statement on recent coding issue. (2022, August 22). Equifax. https://www.equifax.com/newsroom/all-news/-/story/equifax-statement-on-recent-coding-issue/

Finance Train. (n.d.). Handling missing values in time series. Finance Train. https://financetrain.com/handling-missing-values-in-time-series

Hugging Face. (n.d.-a). Hugging face hub documentation. Hugging Face Hub documentation. https://huggingface.co/docs/hub/index

IBM. (n.d.). Infosphere qualitystage report templates. https://www.ibm.com/docs/en/iis/11.5?topic=templates-infosphere-qualitystage-report

IBM Research. (n.d.). What is retrieval-augmented generation (RAG)? IBM. https://research.ibm.com/blog/retrieval-augmented-generation-RAG?ref=robkerr.ai

Informatica. (n.d.). Replace missing values. https://docs.informatica.com/data-quality-and-governance/informatica-data-quality/10-4-0/developer-transformation-guide/sequence-generator-transformation/sequence-generator-ports/nextval-port/replace-missing-values.html

Kober, R., Subraamanniam, T., & Watson, J. (2011). The impact of total quality management adoption on small and medium enterprises' financial performance. Accounting & Finance, 52(2), 421–438. https://doi.org/10.1111/j.1467-629x.2011.00402.x

Kumar, S. (Ed.). (2023). Data integrity and data governance. IntechOpen. https://doi.org/10.5772/intechopen.100778

Martineau, K. (2023, August 23). What is retrieval-augmented generation? IBM. https://research.ibm.com/blog/retrieval-augmented-generation-RAG?ref=robkerr.ai

Microsoft. (n.d.). Temporal table system consistency checks - SQL server. Temporal Table System Consistency Checks - SQL Server | Microsoft Learn. https://learn.microsoft.com/en-us/sql/relational-databases/tables/temporal-table-system-consistency-checks?view=sql-server-ver16

OpenAI. (2023, March 24). March 20 ChatGPT outage: Here's what happened. OpenAI. https://openai.com/index/march-20-chatgpt-outage/

Pansara, R. (2023). Cultivating data quality to strategies, challenges, and impact on decision-making. International Journal of Management Education for Sustainable Development, 6(6), 24-33. https://ijsdcs.com/index.php/IJMESD/article/view/356/131

Roumeliotis, K.I., Tselikas, N.D., & Nasiopoulos, D.K. (2023). Llama 2: Early adopters' utilization of Meta's new open-source pretrained model. Preprints, 2023072142. https://doi.org/10.20944/preprints202307.2142.v2

Sakpal, M. (2021, July 14). 12 actions to improve your data quality. Gartner. https://www.gartner.com/smarterwithgartner/how-to-improve-your-data-quality

Sakpal, M. (n.d.). 12 actions to improve your data quality. Gartner. https://www.gartner.com/smarterwithgartner/how-to-improve-your-data-quality

Strong, D. M., Lee, Y. W., & Wang, R. Y. (1997). Data quality in context. Communications of the ACM, 40(5), 103–110. https://doi.org/10.1145/253769.253804

Talend. (n.d.). Uniques/duplicates. Talend Studio User Guide Help. https://help.talend.com/en-US/studio-user-guide/8.0/uniques-duplicates

Takyar, A. (n.d.). Generative AI for compliance. LeewayHertz. https://www.leewayhertz.com/generative-ai-for-compliance/

Taylor, P. (2023, November 23). Amount of data created, consumed, and stored 2010-2020, with forecasts to 2025. Statista. https://www.statista.com/statistics/871513/worldwide-data-created/

# APPENDICES

## APPENDIX 1: Data Sources

| Source Title | Author | Publication Date | URL |
|---|---|---|---|
| Bluecopa | Srividhya Gurumurthi | December 19, 2023 | https://www.bluecopa.com/blog/finance-data-cleansing |
| Mosaic | Brad Mundell | December 21, 2021 | https://www.mosaic.tech/post/clean-organized-financial-data |
| Dataladder | Ehsan Elahi | July 4, 2022 | https://dataladder.com/how-to-improve-data-quality-in-financial-services/ |
| Avanade | Juliet Rustom, Miles Reah | July 25, 2023 | https://www.avanade.com/en/blogs/avanade-insights/data-analytics/esg-reporting-challenges-financial-services |
| boldbi | Faith Akinyi Ouma | May 8, 2024 | https://www.boldbi.com/blog/from-chaos-to-clarity-mastering-data-standardization |
| Medium | Ronald Wahome | Aug 18, 2018 | https://towardsdatascience.com/cleaning-financial-time-series-data-with-python-f30a3ed580b7 |
| gocardless | Abílio Rodrigues | Mar 2023 | https://gocardless.com/guides/posts/open-banking-data-cleaning-and-enrichment/ |
| hqsoftwarelab | Andrey Kazakevich | October 31, 2023 | https://hqsoftwarelab.com/blog/challenges-of-ai-in-fintech/ |
| hqsoftwarelab | Andrey Kazakevich | December 21, 2023 | https://hqsoftwarelab.com/blog/ai-and-ml-in-fintech-transforming-financial-services/ |
| flagrigh | Joseph Ibitola | August 30, 2023 | https://www.flagright.com/post/data-standardization-for-effective-compliance-reporting |