PRN NO: 2019016400827507 Roll No**:** 219503

1. Name of the Student: KEDARE SHIVAM SUNIL

1. Title of the Project:

STRING-MAPPING USING PYTHON FUZZY LOGIC

1. Name of the Guide:

Asst. Prof. P.R. SABALE

1. Teaching experience of the Guide: Three Year
2. Is this your first submission? Yes No

Signature of the Student Signature of the Guide

Date: ………………… Date: ………………

Signature of the Coordinator Date: …………………

##### String-Mapping using Python Fuzzy Logic

##### A Project Report

Submitted in partial fulfillment of the Requirements for the award of the Degree of

#### BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

##### By

Name of The Student

* 1. Kedare Shivam Sunil

Seat Number: 219503

**Under the esteemed guidance of**

**Asst. Prof. P.R. Sabale**

#### Assistant Professor



##### DEPARTMENT OF INFORMATION TECHNOLOGY

**JEEVANDEEP SHAIKSHANIK SANSTHA POI'S ARTS, COMMERCE AND SCIENCE COLLEGE, KHARDI**

***(Affiliated to University of Mumbai)* KHARDI, PIN CODE - 421602 MAHARASHTRA**

##### YEAR 2021-2022

**JEEVANDEEP SHAIKSHANIK SANSTHA POI'S ARTS, COMMERCE AND SCIENCE COLLEGE, KHARDI**

***(Affiliated to University Of Mumbai)***

**CITY-MAHARASHTRA PINCODE-421602**

#### DEPARTMENT OF INFORMATION TECHNOLOGY



**CERTIFICATE**

This is to certify that the project entitled, **"STRING-MAPPING USING PYTHON FUZZY LOGIC"**, is bonafied work of Kedare Shivam bearing Seat. No: (**219503**) submitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY from University of Mumbai.

##### Internal Guide Coordinator

**External Examiner**

**Date: College Seal**

**Abstract**

This work presents probabilistic approach to the problem of schema mapping. Our approach is declarative, scalable, and extensible. It builds upon recent results in both schema mapping and probabilistic reasoning and contributes novel techniques in both fields. We introduce the problem of mapping selection, that is, choosing the best mapping from a space of potential mappings, given both metadata constraints and a data example. As selection has to reason holistically about the inputs and the dependencies between the chosen mappings, we define a new schema mapping optimization problem which captures interactions between mappings. Using hundreds of realistic integration scenarios, we demonstrate that the accuracy of Combined Mapping Approach is more than 33% above that of metadata-only approaches already for small data examples, and that Combined Mapping Approach routinely finds perfect mappings even if a quarter of the data is inconsistent.

# ACKNOWLEDGEMENT

Before we get into thick of things, I would like to add few heartfelt words for the people who are part of our team as they have been unending contribution right from the start of construction of the report. Apart from the team I am indebted to the numbers of persons who have provided helpful and constructive guidance in the draft of material. I acknowledge with deep sense of gratitude towards the encouragement in the form of substantial assistance provided each and every member of my team. I would like to extend my sincere thanks to our guide **Asst.Prof. P.R.Sabale**

It is my great fortune that I have got opportunity to carry out this project work under the supervision of **Prof. Pushpa Sable**, in the **Department of Bachelor of information Tecnology (BscIT**), **JSSP College Khardi**, affiliated to **Mumbai University of Information Technology** Maharashtra, India. I express my sincere thanks and deepest sense of gratitude to my guide for his constant support, unparalleled guidance and limitless encouragement. I wish to convey my gratitude to Prof. **Pushpa Sable**, **HOD, Department of Information Technology**, and for providing all kinds of infrastructural facility towards the research work. I would also like to convey my gratitude to all the faculty members and staffs of the Department of information Technology, for their whole hearted cooperation to make this work turn into reality.

# DECLARATION

I hereby declare that the project entitled, “**String-Mapping using Python Fuzzy Logic**” done at **Khardi**, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of

#### BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY) to be

Submitted as final semester project as part of our curriculum.

##### Name and Signature of the Student

##### Kedare Shivam Sunil

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Contents** | **Page No.** |
| **1.** | **CHAPTER 1: INTRODUCTION**   * 1. Background   2. Objectives   3. Purpose and Scope Applicability      1. Purpose      2. Scope      3. Applicability | 11  13  14  14  15 |
| **2.** | **CHAPTER 2: SURVEY OF TECHNOLOGIES**   * 1. Existing System   2. Proposed System   3. Software Requirements   4. Justification of platform |  |
| **3.** | **CHAPTER 3: SYSTEM DESIGN**   * 1. Basic Modules   3.2Architecture Design  3.3 System Development  3.4 System Flow |  |
| **4.** | **CHAPTER 4: IMPLEMETATION AND TESTING** |  |
| **6.** | **CHAPTER 6: CONCLUSION AND FUTURE WORK** |  |
| **7.** | **CHAPTER 7: REFERNCES** |  |

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Sr No** | **Figure name** | **Page No** |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |
| 6 |  |  |
| 7 |  |  |
| 8 |  |  |
| 9 |  |  |

**Chapter 1 Introduction**

Data mapping is the technique of identifying objects which are related to each other. In other words, database mapping is a method of finding the correspondences between the concepts of different distributed, heterogeneous data sources. A database mapper is a binary operation which takes two schemas as input and determines their corresponding elements. Generally, the determination of all matches between elements of two schemas is not possible due to the existence of semantic in input schemas. Therefore, a database mapper should only provide match candidates through a user interface in such a way that users can accept, reject, or change them. The several semi-automatic database mapping algorithms have been proposed to determine matches between different elements of two schemas [1][2][3].

Enterprise data is getting more diverse and at the same time, it is advantageous for businesses to leverage data and transform it into meaningful results. However, enterprises today collect information from many sources referring the same entities. To merge this data and make sense of it, data mapping is used which is the process of establishing relationships between separate database schemas.

For example, Microsoft Dynamics CRM contains several data sets which comprise of different objects, such as Leads, Opportunities, and Competitors. Each of these data sets has several fields like Name, Account Owner, City, Country, Job Title, and more. The application also has a defined schema along with attributes, enumerations, and mapping rules. Therefore, if a new record is to be added to the schema of a data object, a data map needs to be created from the data source to the Microsoft Dynamics CRM account.

Depending on the number, schema, and primary keys and foreign keys of the relational databases data sources, database mappings can have a varying degree of complexity. For example, in the following example, data from three different databases tables are joined and mapped to an Excel destination. Thus, attribute level information and similar entities are to be tackled for the effective database mapping [4][5].

Talking about the simple language, data mapping is a relationship between two data systems. Data mapping connects two different types of data models together. In data integration process data mapping is one of the most important factors. Making it simpler, data mapping is all about finding how a computer application or database connects to another computer or database. To make it easier and simpler to understand the concept, here is an example.

Suppose you have a list of people, a list that contains a few names of your friends. As well you have another list that contains the phone numbers of your friends. Now you want to connect both of the lists together and that is where data mapping works for you.

Data mapping is the first step in many complex sectors that come under the data integration process. That includes data transformation between the data source and the data destination. It helps in the discovery of a private data such as the last digits in a social security number and so on. It can also be used for securing many databases and merge them into one while looking for redundancy [6].

Probabilistic record linkage, sometimes called fuzzy matching (also probabilistic merging or fuzzy merging in the context of merging of databases), takes a different approach to the record linkage problem by taking into account a wider range of potential identifiers, computing weights for each identifier based on its estimated ability to correctly identify a match or a non-match, and using these weights to calculate the probability that two given records refer to the same entity. Record pairs with probabilities above a certain threshold are considered to be matches, while pairs with probabilities below another threshold are considered to be non-matches; pairs that fall between these two thresholds are considered to be "possible matches" and can be dealt with accordingly (e.g., human reviewed, linked, or not linked, depending on the requirements). Whereas deterministic record linkage requires a series of potentially complex rules to be programmed ahead of time, probabilistic record linkage methods can be "trained" to perform well with much less human intervention [11].

* 1. **Background**

In this section, we discuss the related work and motivate our study about ‘String-Mapping using Python Fuzzy Logic’.

Data professionals use data mapping to assist in three main areas:

(a) [Data Integration](https://www.xplenty.com/blog/whats-a-data-warehouse-why-are-they-important-and-what-technology-should-a-modern-data-warehouse-inc/)

(b) [Data Migration](https://www.xplenty.com/blog/data-transformation-explained/)

(c) [Data](https://www.xplenty.com/blog/data-migration-simplified-understanding-the-use-cases-challenges-and-processes-of-data-migration/) Transformation.

1. **Data Integration**

The data integration process mainly focuses on new information. What actually date integration process do is it creates a way between a new data model and an old data model and makes them connected to each other. By connecting both of the new and old data models it gets easier to access data from both of the sources. You can say it creates an argument between the new and existing data.

Data integration involves the process of integrating various types of applications across the business ecosystem. It can either be scheduled or can be triggered by an event. Similar to data migration, data maps for integration must also match fields such as source and destination [9].

1. [**Data Migration**](https://www.xplenty.com/blog/data-migration-simplified-understanding-the-use-cases-challenges-and-processes-of-data-migration/)

The data migration process focuses on mainly moving information from one data model to another data model. What mapping actually do is, it creates a way between the source data model and the destination data model. That is where a data mapping software comes to the light. The software makes the migration process really easy and efficient. For example, suppose you have purchased a new mobile and now you want to move all the data that contains contacts number, text messages, and photos to your new mobile from the old one. To perform this task you will simply use a software or a feature like Bluetooth to get the job done. Well, that’s how data migration works [9].

Data migration involves the process of moving data between systems as a one-time event. Followed by the migration process, the destination is the new source of migrated data, and consequently, the source gets eliminated. Data mapping tools can be used to perform a migration process by mapping source data fields to destination fields.

1. [**Data Transformation**](https://www.xplenty.com/blog/data-migration-simplified-understanding-the-use-cases-challenges-and-processes-of-data-migration/)

Data transformation involves the process of transforming data from a source format into a destination format. Companies can use data transformation software to cleanse data, eliminating nulls or duplicates, collecting data, enriching the data, or other transformations.

If the objective is to transfer all the data into one source called warehouse for further purpose. As the users run a query or analysis, data is fetched from a warehouse. Data that is present in the warehouse has already undergone migration, integration, and transformation. Data mapping tools allow organizations to ensure that as data comes into the warehouse, it gets to its destination the way it was intended.

* 1. **Objectives**

Source to target mappings describe how one or more attributes in source data sets are related to one or more attributes in a target data set. These source-to-target mappings are derived from the ETL transformation rules described in a requirements specification document, comments inside the transformation scripts, spreadsheets, ER diagrams, or SQL scripts. Data mapping is complex and challenging. So, what makes data mapping so difficult? The challenges and shortcomings associated with data mapping and how they can be mitigated.

* 1. **Purpose and Scope Applicability**
     1. **Purpose**

The process of connecting data sources, building mappings for data transformation and integration, and validating the transformed data often require significant resources, particularly when the entire process is done manually. There are several ways to ease the data mapping burden significantly. It starts by defining the process for gathering information to be documented for each source and target. In most cases, systematic interviews with data stewards are the most efficient way to collect info for a data map. Interviews with subject matter experts (SME’s) should be direct using data mapping templates. Meta data and data mapping tools should be used to automate as much as possible.

* + 1. **Scope**

Creating manual mappings using spreadsheets is often difficult and time-consuming. Mapping’s specifications built using spreadsheets cannot be easily managed Data mappings cannot be easily versioned and auditability of what and who has changed mappings remains a constant issue.

Creating maps internally and using unqualified personnel for map development compromises the integrity of results. Use skilled personnel familiar with data mapping requirements, limitations, and pitfalls to ensure reliable results [9].

* + 1. **Applicability**

1. **Transparency for analysts and architects**

Since [data quality](https://www.talend.com/resources/data-quality/) is important, data analysts and architects need a precise, real-time view of the data at its source and destination. Data mapping tools provide a common view into the data structures being mapped so that analysts and architects can all see the data content, flow, and transformations [9].

1. **Optimization of complex formats**

With so much data streaming from diverse sources, data compatibility becomes a potential problem. Good data mapping tools streamline the [transformation process](https://www.talend.com/resources/data-transformation-defined/) by providing built-in tools to ensure the accurate transformation of complex formats, which saves time and reduces the possibility of human error [9].

1. **Fewer challenges for changing data models**

Data maps are not a one-and-done deal. Changes in [data standards, reporting requirements, and systems](https://www.talend.com/resources/what-is-data-governance/) mean that maps need maintenance. With a cloud-based data mapping tool, stakeholders no longer run the risk of losing documentation about changes. Good data mapping tools allow users to track the impact of changes as maps are updated. Data mapping tools also allow users to reuse maps, so you don't have to start from scratch each time [9].

### **Master data management**

Most [Master data management](https://en.wikipedia.org/wiki/Master_data_management) (MDM) products use a record linkage process to identify records from different sources representing the same real-world entity. This linkage is used to create a "golden master record" containing the cleaned, reconciled data about the entity. The techniques used in MDM are the same as for record linkage generally. MDM expands this matching not only to create a "golden master record" but to infer relationships also. (i.e., a person has a same/similar surname and same/similar address, this might imply they share a household relationship) [10].

### **Data warehousing and business intelligence**

Record linkage plays a key role in [data warehousing](https://en.wikipedia.org/wiki/Data_warehousing) and [business intelligence](https://en.wikipedia.org/wiki/Business_intelligence). Data warehouses serve to combine data from many different operational source systems into one [logical data model](https://en.wikipedia.org/wiki/Logical_data_model), which can then be subsequently fed into a business intelligence system for reporting and analytics. Each operational source system may have its own method of identifying the same entities used in the logical data model, so record linkage between the different sources becomes necessary to ensure that the information about a particular entity in one source system can be seamlessly compared with information about the same entity from another source system. Data standardization and subsequent record linkage often occur in the "transform" portion of the [extract, transform, load](https://en.wikipedia.org/wiki/Extract,_transform,_load) (ETL) process [10].

### **Historical research**

Record linkage is important to social history research since most data sets, such as [census records](https://en.wikipedia.org/wiki/Census) and parish registers were recorded long before the invention of [National identification numbers](https://en.wikipedia.org/wiki/National_identification_number). When old sources are digitized, linking of data sets is a prerequisite for [longitudinal study](https://en.wikipedia.org/wiki/Longitudinal_study). This process is often further complicated by lack of standard spelling of names, family names that change according to place of dwelling, changing of administrative boundaries, and problems of checking the data against other sources. Record linkage was among the most prominent themes in the [History and computing](https://en.wikipedia.org/w/index.php?title=History_and_computing&action=edit&redlink=1) field in the 1980s, but has since been subject to less attention in research [10].

### **Medical practice and research**

Record linkage is an important tool in creating data required for examining the health of the public and of the health care system itself. It can be used to improve data holdings, data collection, quality assessment, and the dissemination of information. Data sources can be examined to eliminate duplicate records, to identify under-reporting and missing cases (e.g., census population counts), to create person-oriented health statistics, and to generate disease registries and health surveillance systems. Some cancer registries link various data sources (e.g., hospital admissions, pathology and clinical reports, and death registrations) to generate their registries. Record linkage is also used to create health indicators. For example, fetal and infant mortality is a general indicator of a country's socioeconomic development, public health, and maternal and child services. If infant death records are matched to birth records, it is possible to use birth variables, such as birth weight and gestational age, along with mortality data, such as cause of death, in analyzing the data [10].

**Chapter - 02**

**SURVEY OF TECHNOLOGIES**

The Survey Of Technologies some schema matching techniques like complex schema matching, Entity matching, mapping between different elements of two schemas, record matching, probabilistic approach of schema mapping, Automatic schema matching are discussed below.

**2.1 Existing System**

The various important approaches of database mapping are discussed as follows:

CSM is complex schema matching, the problem of finding semantic correspondences between elements of two complex schemas, plays a key role in many applications, such as data warehouse, heterogeneous data sources integration and semantic web. The existing approaches to automating schema matching almost focus on computing direct element matches between two schemas. However, this work involves the relationships between real-world schemas involving many complexes matches besides 1:1 matches. The limitation still remains on matching efficiency, because the candidate matches space is so large which they need searching [1].

Entity matching and value mapping across two heterogeneous information sources are critical tasks in applications involving data integration, data warehousing, and federation of databases. Before data can be integrated from multiple tables, the columns and the values appearing in the tables must be matched. Anuj Jaiswal, David J. Miller proposed a novel method that optimizes embedded value mappings to enhance entity matching in the presence of opaque data values and column names. In this approach, the fitness objective for matching a pair of attributes from two entities depends on the value mapping function for each of the two attributes [2].

The work includes schema matching, which provides a mapping between different elements of two schemas. It is complicated due to the existence of semantic in input schemas. It is achieved through different matchers for merging databases. It determines all matches between elements of two schemas. Still, it there is scope for improving accuracy [3].

The schema matching and record matching are two necessary steps in integrating multiple relational tables of different schemas, where it first unifies the schemas and then it detects records referring to the same real-world entity. The two processes have been thoroughly studied separately, but in this work, they had done combination of both. They have used indexing and estimated the matching likelihood between two records.[4]

To link the data stored in heterogeneous data sources, a critical problem is entity matching, i.e., matching records representing semantically corresponding entities in the real world, across the sources. While decision tree techniques have been used to learn entity matching rules, most decision tree learners have an inherent representational bias, that is, they generate unvaried trees and restrict the decision boundaries to be axis-orthogonal hyper-planes in the feature space. Cascading other classification methods with decision tree learners can alleviate this bias and potentially increase classification accuracy. In this paper, the authors apply a recently-developed constrained cascade generalization method in entity matching and report on empirical evaluation using real-world data. The evaluation results show that this method outperforms the base classification methods in terms of classification accuracy, especially in the dirtiest case [5].

The probabilistic approach of schema mapping is declarative, scalable, and extensible. It builds upon recent results in both schema mapping and probabilistic reasoning novel techniques in both fields. The problem of mapping selection, that is, choosing the best mapping from a space of potential mappings, given both metadata constraints and a data example. As selection has to reason holistically about the inputs and the dependencies between the chosen mappings a new schema mapping optimization problem which captures interactions between mappings then they introduce Collective Mapping Discovery (CMD), solution to this problem using state of- the-art probabilistic reasoning techniques, which allows for inconsistencies and incompleteness [6].

Automatic schema matching algorithms are typically only concerned with finding attribute correspondences. However, real world data integration problems often require matchings whose arguments span all three types of elements in relational databases: relation, attribute and data value. This paper introduces the definitions and semantics of three additional correspondence types concerning both schema and data values. Two methods for automatically identifying these correspondences are developed. One requires a limited number of duplicates across data sources. The other is a general instance-based method with no such requirement. Experiments conducted on four real world data sets demonstrate the effectiveness of the methods [7].

The following table shows the comparative analysis of various Schema Mapping techniques which is discussed in above literature review.

**Table 1: Comparison of Schema Matching Techniques**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr No.** | **METHOD** | **DESCRIPTION** | **BENEFITS** | **LIMITATIONS** |
| **1.** | **Complex Schema Matching** | It can filter unreasonable matches on data types and values by preprocessor and clustering processor and employs a set of special-purpose searchers in match generator to explore a specialized portion of the search space and discovers 1:1 and complex matches. Then it estimates candidate matches and selects optimal candidate matches by using similarity estimator and match selector respectively. | In CSM, exploiting domain knowledge brings even  greater benefits, because it can also help to direct the search process and prune meaningless candidate early, avoiding costly evaluation. CSM besides has exploits domain constraints and past matches, it also exploits  overlap data between the databases and external data in the domain. | There are opaque columns in the schemas  being matched, it can apply complementary matcher to  discover matching relations between opaque columns |
| **2.** | **Uninterpreted Schema Matching** | This method optimizes embedded value mappings to enhance schema matching in the presence of opaque data values and column names. In this approach, the fitness objective for matching a pair of attributes from two schemas depends on the value mapping function for each of the two attributes. | Uninterpreted schema matching methods should form a useful addition to a suite of (semi) automated tools for resolving structural heterogeneity. | If two columns to be matched do, in principle, correspond to the  same random variable, there may be some hidden factor  that differs for the two schemas and that will cause the two columns that ground-truth are indeed a matching pair to differ in their probability mass functions |
| **3.** | **Automatic Domain Independent Schema Matching** | Automatic Domain Independent Schema matching approach  which utilized both the structural and semantic information during the process of schema  matching and offered database integration without user intervention. | Automatic Domain Independent Schema matching approach  produced better global schemas during integration process. | In these Schema matching it is observed that without user intervention, i.e. automating the process of schema integration,  applying different types of matchers produce different global schemas. |
| **4.** | **Supervised Instance Matching** | ScLink, an instance matching system that can generate a configuration automatically.  In ScLink, two novel supervised learning algorithms, cLearn and minBlock are used. cLearn applies an apriori-like heuristic for finding the optimal combination of matching properties and similarity metrics. minBlock finds a blocking model, which aims at optimally reducing the pairwise alignments of instances between input repositories. | On standard and very large datasets find that minBlock and  cLearn are very effective and efficient. cLearn is also significantly better than existing configuration  learning algorithms. It drastically boosts the accuracy of ScLink and makes the  system outperform the state-of-the-arts, even when being trained using a small amount of  labeled data. | In Supervised Instance Matching issue of identifying exactly the equivalent properties between the trained repositories and the unknown  repositories having different schema remains unsolved. |
| **5.** | **Probabilistic Schema Mapping** | Probabilistic Schema Mapping is declarative, scalable, and  extensible. It builds upon recent results in both schema mapping  and probabilistic reasoning and contributes novel techniques in  both fields. They introduced the problem of mapping selection,  that is, choosing the best mapping from a space of potential  mappings, given both metadata constraints and a data example. | These techniques are routinely used to combine logical constraints in relational domains with the ability to handle uncertainty and conflicting information. | The source selection problem has been modeled as a problem of finding a set of sources. |
| **6.** | **Automatic Schema Matching** | Automatic schema matching algorithms are typically only concerned with finding attribute correspondences. However, real world data integration problems often require matchings whose arguments span all three types of elements in relational databases: relation, attribute and data value. | These compound correspondences  complement the typical attribute correspondences.  The correspondences can be formulated to tgds, and  used in data exchange and data integration applications. | 1) Ranking algorithm of duplicate pairs considering  diversity in addition to similarity is important to improve the recall  of duplicate methods.  2) Measures for non-string  data types are a promising direction to improve schema matching  systems. |

**2.2 Proposed** **System**

In this proposed system, combined database mapping approach for interoperating among different database systems. In this proposed system, the database mapping process uses of three main phases: -attribute mapping, data type mapping and data merging.

The attribute mapping determines whether the names of two attributes are similar or not by calculating the percentage of similarities between their names. It uses both structural and semantic level information including string mapping, dictionaries to calculate the percentage of similarities between the names of two attributes. Then, our approach compares the similar mapping attributes and data type mapping and selects the better match. The final phase of the attribute mapping process is merging, which merges those pairs of attributes that are identified matched, while those attributes that are not matched remain as they are.

**Block Diagram**

User

Merging Data

Data type mapping

Attribute mapping

**Fig. 2.1 Conceptual Model**

**2.2.1 User**

The users basically include engineers, scientist, business analytics and others who thoroughly familiarize themselves with the facilities of the concept in order to implement their application to meet their complex requirement. These users try to learn most of the Data mapping facilities in order to achieve their complex requirements.

**2.2.2 Merging Data**

Data merging is a method for merging similar datasets from two or more tables to create a single data set (table) for easy reporting & analysis.

Let us say for instance you have a Sales database that contains yearly sales data stored as individual tables (like Sales 2016, Sales 2017 etc). What if you want to merge the "Sales 2016" data along with the "Sales 2017" data and create a consolidated table such that you can analyze the sales across all the years?  This is where the data merging feature comes in handy. You can merge the data from any number of tables and store it as a single table for reporting and analysis.

**2.3 Hardware & Software Requirements**

**Software**

* Python 3.0 or latest version
* Anaconda Environment latest version
* Packages:
* To Install recordlinkage : conda install -c conda-forge recordlinkage
* To Install fuzzymatcher : conda install -c conda-forge fuzzymatcher
* Juypter Notebook 6.1.4

**Hardware**

* Processor- Intel(R) Core(TM) i3-7020U CPU@ 2.30GHz
* RAM- 8 GB
* System Type – 64-bit Operating System

**2.4 Justification of Platform**

* **The time, people, and tools needed to build data maps can be substantial**

The process of connecting data sources, building mappings for data transformation and integration, and validating the transformed data often require significant resources, particularly when the entire process is done manually.

There are several ways to ease the data mapping burden significantly. It starts by defining the process for gathering information to be documented for each source and target. In most cases, systematic interviews with data stewards are the most efficient way to collect info for a data map. Interviews with subject matter experts (SME’s) should be direct using data mapping templates. Meta data and data mapping tools should be used to automate as much as possible.

* **The information needed is not always available for building data maps**

A common mistake organization make with data maps is that they omit important information and therefore render the data map far less useful than it should be. Before data mapping initiatives get off the ground, project organizers should assemble key stakeholders and gather feedback on what information needs to be included in mappings for sources and targets. For example, retention schedules, litigation risk profiles, and accessibility constraints of particular data sources. Privacy officers will want to know which data sources contain sensitive customer information that must be carefully protected.

* **Substantial efforts needed to maintain data maps**

As with all important project documents, data maps should be constantly evaluated, updated and assessed for quality. One method to ensure the data mappings are maintained is to make sure the process is fully integrated into the organization’s master data management program. With every change to requirements, data maps should be reviewed to assess the impact.

* **Data mapping with spreadsheets can pose long-term issues**

Creating manual mappings using spreadsheets is often difficult and time-consuming. Mapping’s specifications built using spreadsheets cannot be easily managed Data mappings cannot be easily versioned and auditability of what and who has changed mappings remains a constant issue.

Creating maps internally and using unqualified personnel for map development compromises the integrity of results. Use skilled personnel familiar with data mapping requirements, limitations, and pitfalls to ensure reliable results [9].

**CHAPTER -03**

**SYSTEM DESIGN**

System design is the phase that bridges the gap between problem domain and the existing system in a manageable way. This phase focuses on the solution domain, i.e. “how to implement?”

In this phase, the complex activity of system development is divided into several smaller sub-activities, which coordinate with each other to achieve the main objective of system development.

The flowchart for the proposed system is shown in Fig.4.1 as follow.

Start

Stop

Display Data

Attribute

Mapped?

Apply Mapping Algorithm

Apply the Fuzzy Mapping

Select the Attribute for Mapping

Input Schema

No

Yes

**Fig 4.1 -** **Flowchart for The Proposed System**

**3.1 System Development**

* **Input Schema**

# In this step first choose the two relational dataset that you need to mapped. After that import selected datasets by read\_csv( ) function. Opening a CSV file through this is easy. But there are many others thing one can do through this function only to change the returned object completely. For instance, one can read a csv file not only locally, but from a URL through read\_csv or one can choose what columns needed to export [13].

* **Selection of Attribute**

In this step attribute is selected from each chose datasets but the columns have different names, we need to define which columns to match for the left and right Data Frames and then fuzzy matcher try to figure out the matches.

* **Fuzzy Mapping**

Now let fuzzy matcher try to figure out the matches by using fuzzy\_left\_join , it allows the user to fuzzy match two pandas dataframes based on one or more common fields. The various comparison functions are discussed below.

1. **DataPreprocessor**

A DataPreprocessor is responsible for ingesting df\_left (the dataframe containing the records we want to find matches for) and df\_right (the dataframe we want to search for potential matches) and applying preprocessing stages like normalization to make matching easier.

1. **DataGetter**

A DataGetter handles the retrieval of data from df\_right (the dataframe in which to search for matches) It retrieves a list of potential match ids.

A set of informative, discriminating and independent features is important for a good classification of record pairs into matching and distinct pairs. The [recordlinkage.Compare](https://recordlinkage.readthedocs.io/en/latest/ref-compare.html" \l "recordlinkage.Compare" \o "recordlinkage.Compare) class and its methods can be used to compare records pairs.

Several comparison methods are included such as string similarity measures, numerical measures and distance measures [12].

1. **Compare ( )**

Recordlinkage to compare record pairs with efficiently. It is used to compare the attributes of candidate record pairs. The Compare class has methods like string, exact and numeric to initialize the comparing of the records. The compute method is used to start the actual comparing.

**Syntax:**

classrecordlinkage.Compare(features= [], n\_jobs=1, indexing\_type='label', \*\*kwargs)

1. **Compute ()**

Compare the records of each record pair. Calling this method starts the comparing of records.

**Syntax:** compute (pairs, x, x\_link=None)

Parameters:

1. pairs ([pandas.MultiIndex](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.MultiIndex.html" \l "pandas.MultiIndex" \o "(in pandas v0.25.3))) – A pandas MultiIndex with the record pairs to compare. The indices in the MultiIndex are indices of the DataFrame(s) to link.
2. x ([pandas. DataFrame](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html#pandas.DataFrame)) – The DataFrame to link. If x\_link is given, the comparing is a linking problem. If x\_link is not given, the problem is one of duplicate detection.
3. x\_link ([pandas.DataFrame](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html" \l "pandas.DataFrame" \o "(in pandas v0.25.3)), optional) – The second DataFrame.
4. **compare\_vectorized( )**

Compute the similarity between values with a callable. This method initializes the comparing of values with a custom function/callable. The function/callable should accept numpy.ndarray’s.

**Syntax:** compare\_vectorized (comp\_func, labels\_left, labels\_right, \*args, \*\*kwargs)

Parameters:

1. comp\_func (function) – A comparison function. This function can be a built-in function or a user defined comparison function. The function should accept numpy.ndarray’s as first two arguments.
2. labels\_left (label, [pandas.Series](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.Series.html" \l "pandas.Series" \o "(in pandas v0.25.3)), [pandas.DataFrame](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html" \l "pandas.DataFrame" \o "(in pandas v0.25.3))) – The labels, Series or DataFrame to compare.
3. labels\_right (label, [pandas.Series](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.Series.html" \l "pandas.Series" \o "(in pandas v0.25.3)), [pandas.DataFrame](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html" \l "pandas.DataFrame" \o "(in pandas v0.25.3))) – The labels, Series or DataFrame to compare.
4. \*args – Additional arguments to pass to callable comp\_func.
5. \*\*kwargs – Additional keyword arguments to pass to callable.
6. comp\_func.label ((list of) label(s)) – The name of the feature and the name of the column.
7. **exact ( )**

Compare the record pairs exactly. This is used to compare records in an exact way. The similarity is 1 in case of agreement and 0 otherwise.

**Syntax:** exact(left\_on, right\_on, agree\_value=1, disagree\_value=0, missing\_value=0, label=Non)

Parameters:

1. left\_on ([str](https://docs.python.org/3/library/stdtypes.html#str) or [int](https://docs.python.org/3/library/functions.html#int)) – Field name to compare in left DataFrame.
2. right\_on ([str](https://docs.python.org/3/library/stdtypes.html#str) or [int](https://docs.python.org/3/library/functions.html#int)) – Field name to compare in right DataFrame.
3. agree\_value ([float](https://docs.python.org/3/library/functions.html#float), [str](https://docs.python.org/3/library/stdtypes.html#str), [numpy.dtype](https://docs.scipy.org/doc/numpy/reference/generated/numpy.dtype.html" \l "numpy.dtype" \o "(in NumPy v1.17))) – The value when two records are identical. Default 1. If ‘values’ is passed, then the value of the record pair is passed.
4. disagree\_value ([float](https://docs.python.org/3/library/functions.html#float), [str](https://docs.python.org/3/library/stdtypes.html#str), [numpy.dtype](https://docs.scipy.org/doc/numpy/reference/generated/numpy.dtype.html" \l "numpy.dtype" \o "(in NumPy v1.17))) – The value when two records are not identical.
5. missing\_value ([float](https://docs.python.org/3/library/functions.html#float), [str](https://docs.python.org/3/library/stdtypes.html#str), [numpy.dtype](https://docs.scipy.org/doc/numpy/reference/generated/numpy.dtype.html" \l "numpy.dtype" \o "(in NumPy v1.17))) – The value for a comparison with a missing value. Default 0.

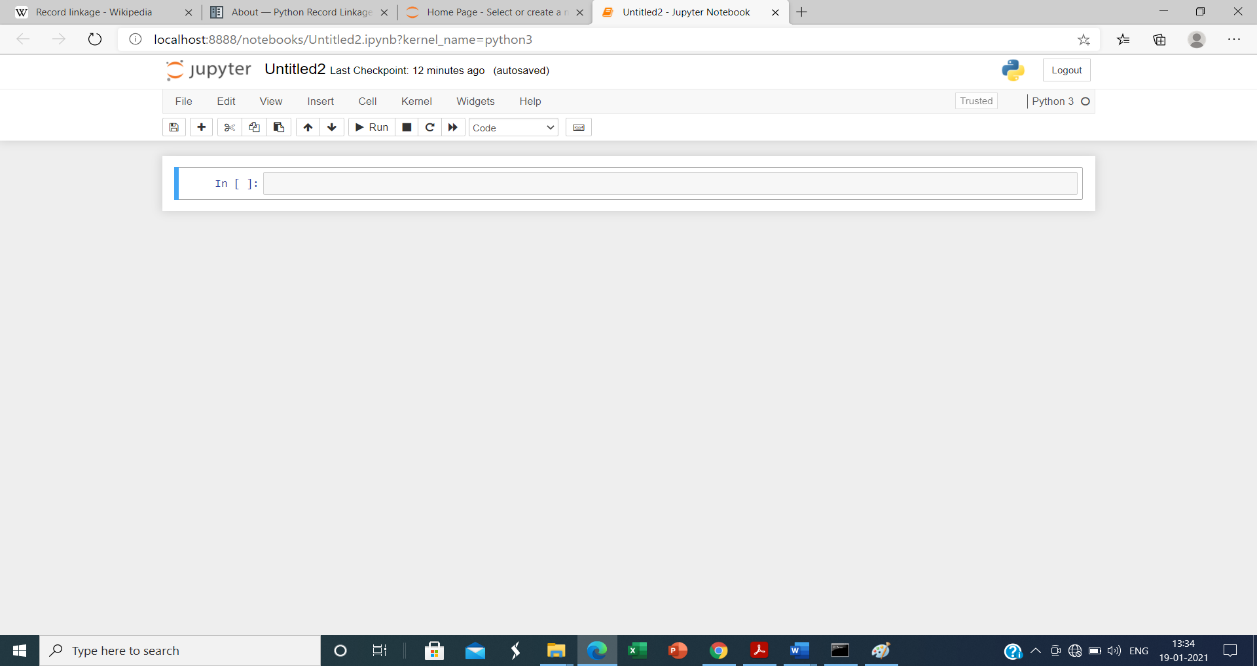
**CHAPTER 4**

**IMPLEMETATION AND TESTING**

**4.1 Implementation Screenshots**

**4.1.1 Jupyter Notebook**

Record linking and fuzzy matching are terms used to describe the process of joining two data sets together that do not have a common unique identifier, so first we open the jupyter notebook from Anaconda Environment.

.

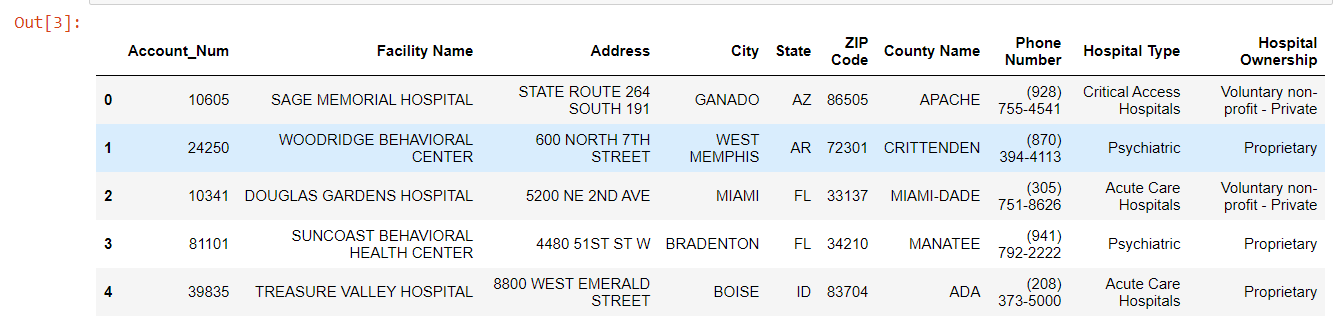
**Figure 4.1 : Jupyter Notebook**

**4.1.2 Datasets**

I chose two hospitals data set because hospital data has some unique qualities that make it challenging to match –

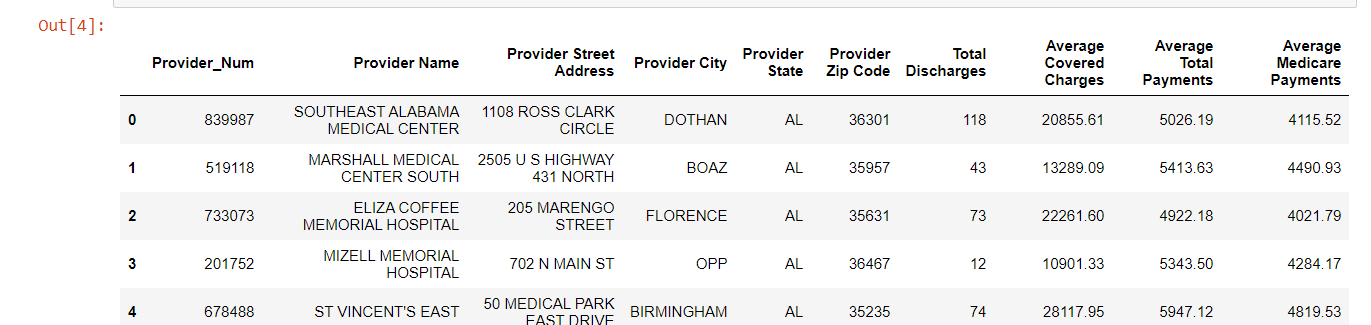
* Many hospitals have similar names across different cities.
* In urban areas, hospitals can occupy several cities blocks so addresses can be ambiguous.
* Hospitals tend to have many clinics and other associated and related facilities nearby.
* Hospitals also get acquired and name changes are common - making this process even more difficult.
* Finally, there are a thousand of medical facilities in the US so the problem is challenging to scale.

The first is an internal data set that contains basic hospital account number, name and ownership information. Here is the hospital account information:



**Figure 4.2: Hospital dataset 1**

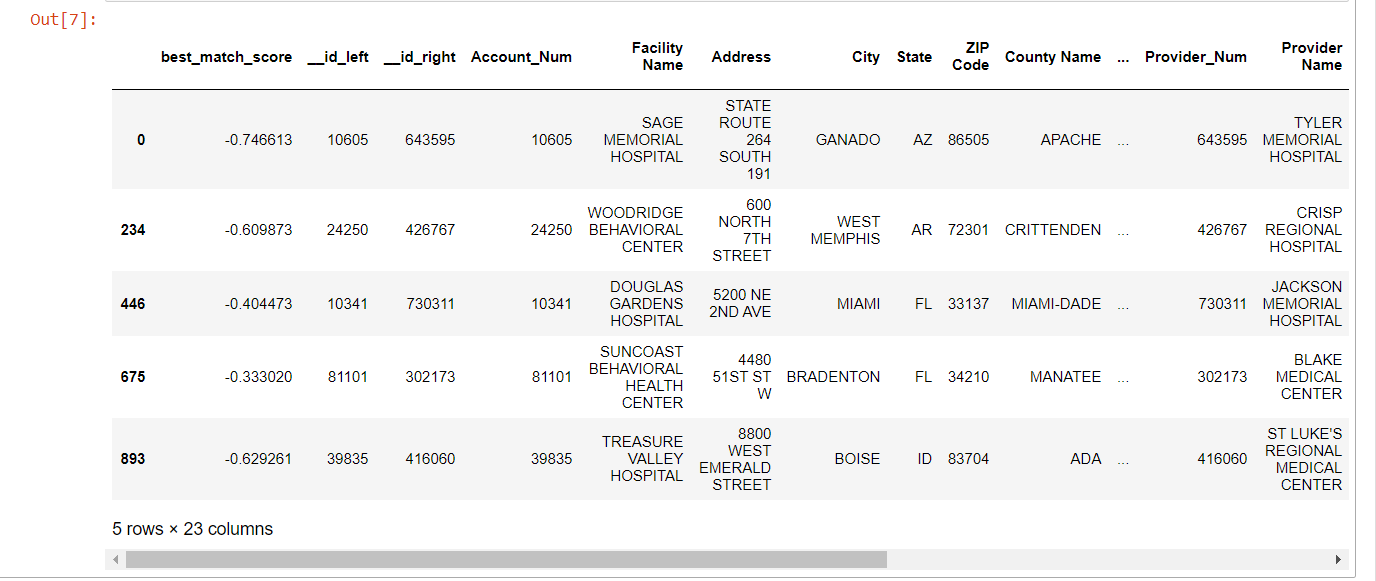
The second data set contains hospital information (called provider) as well as the number of discharges and Medicare payment for a specific Heart Failure procedure. Here is the reimbursement information:



**Figure 4.3 : Hospital dataset 2**

**4.1.3 Fuzzy matcher**

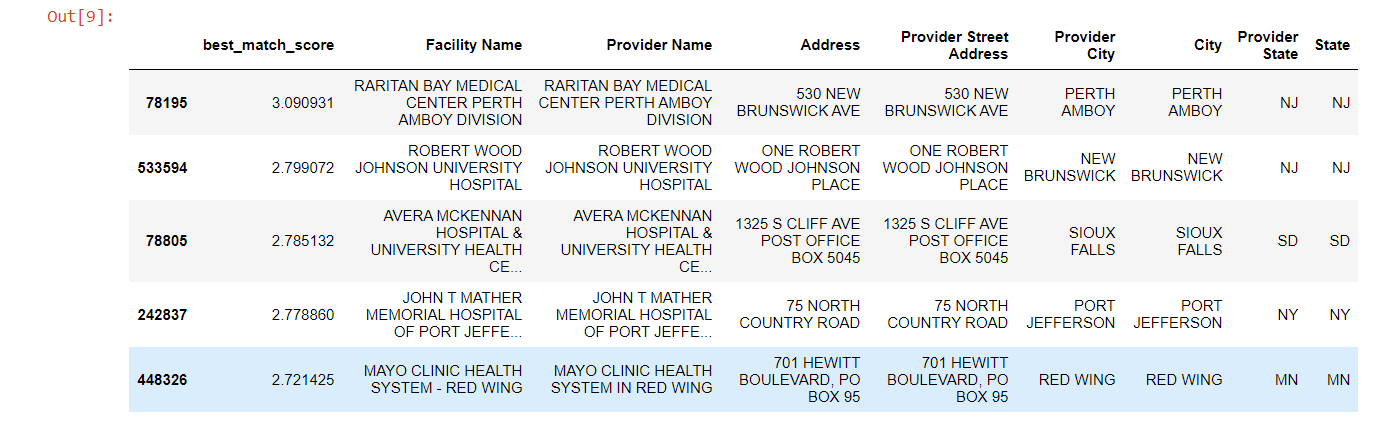
Since the columns have different names, we need to define which columns to match for the left and right Data Frames. In this case, our hospital account information will be the left Data Frame and the reimbursement info will be the right. Now let fuzzy matcher try to figure out the matches using fuzzy\_left\_join.



**Figure 4.4: Fuzzy Matcher**

The matched results Data Frame contains all the data linked together as well as best\_match\_score which shows the quality of the link.

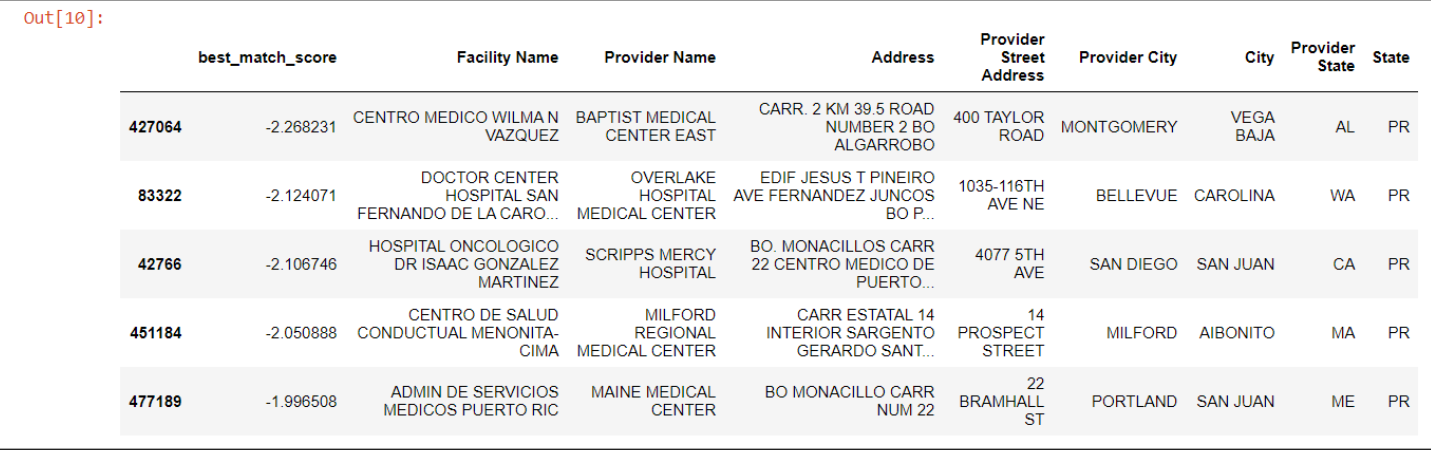
Here’s a subset of the columns rearranged in a more readable format for the top 5 best matches:



**Figure 4.5: Best Matched Results**

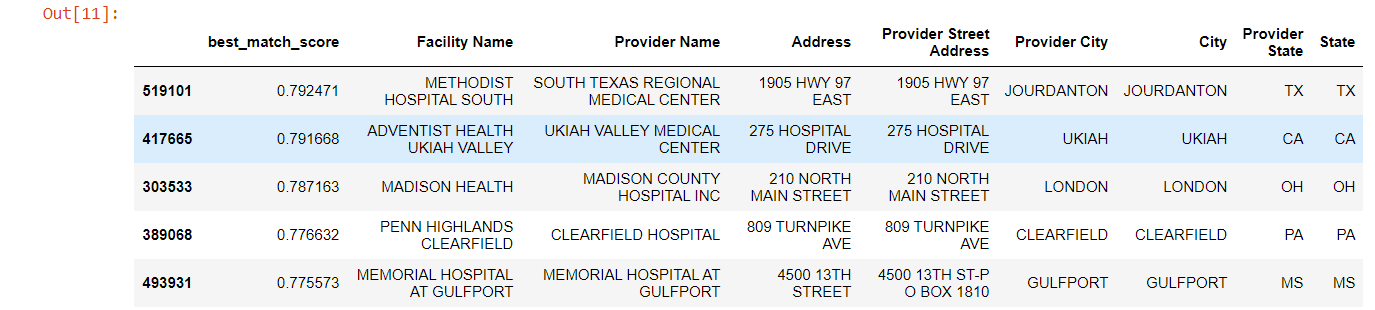
The first item has a match score of 3.09 and certainly looks like a clean match. You can see that the Facility Name and Provider Name for the Mayo Clinic in Red Wing has a slight difference but we were still able to get a good match.

We can check on the opposite end of the spectrum to see where the matches don’t look as good:



**Figure 4.6: Worst Matched Results**

Next here looked at the extreme cases, let’s take a look at some of the matches that might be a little more challenging by looking at scores < 80:



**Figure 4.7: Average Matched Results**

**4.2 Result Analysis**

**4.2.1 Analysis**

Fuzzy matcher determines the best match for each combination. For this data set we are analyzing over 14 million combinations. On the system, this takes about 2 min and 11 seconds to run. The matching is done for three scale best match, worst match & match < 80 in two hospital data set. The result analysis is as follow

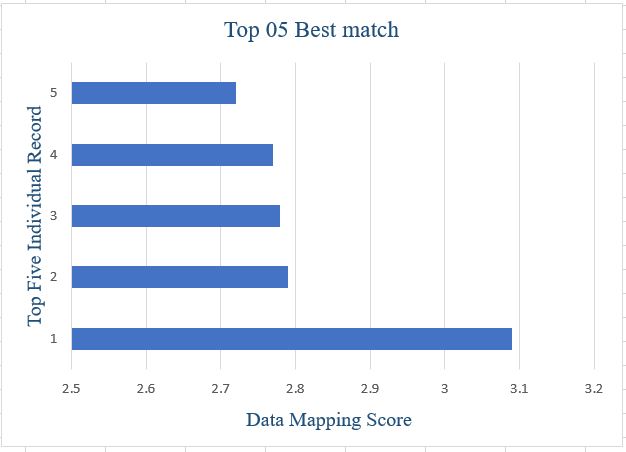
**Table 1- Result Analysis**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Sr no** | **Matching Scale** | **Top Five Match score** | | | | | **Average of 5 match** | **Percentage**  **Match** |
| 1 | Best match | 3.09 | 2.79 | 2.78 | 2.77 | 2.72 | 2.83 | 56.6% |
| 2 | Match < 80 | 0.79 | 0.79 | 0.78 | 0.77 | 0.77 | 0.78 | 15.6% |
| 3 | Worst match | -2.26 | -2.12 | -2.10 | -2.05 | -1.99 | -2.104 | 0 % |

**4.2.2 Observations**

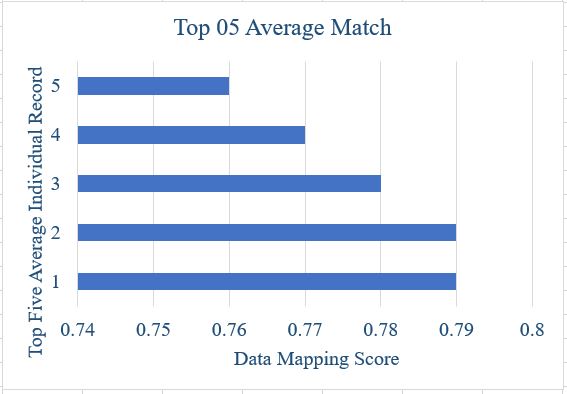
**1) Top Five Best Match**

The Fig.4.2.1 shows the top five best match found in the two-hospital datasets use in the data mapping. In the graph x-axis represent the data mapping score & y-axis represent the top five individual record.



**Fig. 4.2.1 - Best Match in Database**

**2) Top Five Average Match**

. 

**Fig. 4.2.2 - Average Match in Database**

The Fig. 4.2.2 shows the top five Average match found in the two hospital datasets use in the data mapping. In the graph x-axis represent the data mapping score & y-axis represent the top five average case individual record.

The following chart shows the data matching with various scales like best match , average match and worst match

**Fig. 4.2.3 - Various Matching Scale**

**CHAPTER 5**

**CONCLUSION AND FUTURE WORK**

**5.1 Conclusion**

In this work, we had used combined mapping approach for merging databases schemas. Our proposed approach contains structural and semantic level information to map database schemas for getting better results compared to existing approaches.

Data mapping is always resource-intensive requiring hands-on development, review, and knowledge about all sources and targets. Human intervention is necessary for mapping design and validation of map outcomes. Commercial and open-source mapping tools can assist in the process by providing varying degrees of automation. Manual review is required, to a varying extent, to map the portions that failed automated mapping and to validate the results of automated mapping.

Linking different record sets on text fields like names and addresses is a common but challenging data problem. The python ecosystem contains two use full libraries that can take data sets and use multiple algorithms to try to match them together.

Fuzzymatcher uses sqlite’s full text search to simply match two pandas DataFrames together using probabilistic record linkage.

**5.2 Future Scope**

A key to choosing the correct data-mapping solution is product research. Software providers who offer free trail periods make it easier to understand what kind of value is offered. Online reviews may be useful for determining which data mapping programs to investigate further, but business leaders should remember that not every solution is a perfect fit for every user, and some negative reviews may be a result of incompatibility between the users and the software for mapping data. The best data mapping software should be customizable and adaptable enough to provide value to businesses of all kinds.

Some organizations continue documenting data mappings on spreadsheets. However, modern data integrations and migrations are too complex and varied for manual efforts to be effective. With more data, more mappings, and constant changes, such manual processes should be reconsidered. They often lack transparency and don't easily allow tracking the inevitable changes that occur in project requirements, data models, and schemas.

**CHAPTER 6**

**REFERNCES**

* Y.Qian ,Y.Li,J.Song, & L.Yue(2009), “Discovering complex semantic matches between database schemas,” in Proceedings of International Conference on Web Information Systems and Mining, 2009 pages -756-760.
* Anuj Jaiswal, David J. Miller, Member (2010), “Uninterpreted Schema Matching with Embedded Value Mapping under Opaque Column Names and Data Values”, IEEE Transactions on Knowledge and Data Engineering, Vol. 22, No. 2, 2010 pages 291- 309.
* Hamidah Ibrahim, Yaser Mohammad Karasneh, Meghdad Mirabi (2014) “AnAutomatic Domain Independent Schema Matching in Integrating Schemas of Heterogeneous Relational Databases”, Journal of Information Science and Engineering2014 pages -1505-1536.
* Binbin Gu, Zhixu Li, Xiangliang Zhang, A Liu, GuanfengLiu, Kia Zheng, Lei Zhao, Xiaofang Zhou (2017) “The Interaction between Schema Matching and Record Matching in Data Integration”, IEEE Transactions on Knowledge and Data Engineering ,2017pages -186-199.
* Khai Nguyen and Ryutaro Ichise(2016) “ScLink: supervised instance matching system for heterogeneous repositories,” National Institute of Informatics, Vol. 32, 2016, pages -830-863.
* Angelika Kimmig, Alex Memory, Ren´ee J. Miller, Lise Getoor(2017)“ A Collective, Probabilistic Approach to Schema Mapping” in IEEE 33rd International Conference on Data Engineering2017, pages 921-932.
* M.Y. Chaudhari V.M. Shelake (2021) “A Combined Mapping Approach For Relational Schemas” in Vidyabharati International Interdisciplinary Research Journal ISSN 2319-4979 pages 1911-15