Automated Database Schema Matching Engine

Maha Sailan

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

Eastern Mediterranean University
January 2020
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Ali Hakan Ulusoy
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science in Computer Engineering.

Prof. Dr. Işık Aybay
Chair, Department of Computer Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science in Computer Engineering.

Assoc. Prof. Dr. Duygu Çelik Ertuğrul
Supervisor

Examining Committee

1. Assoc. Prof. Dr. Duygu Çelik Ertuğrul

2. Asst. Prof. Dr. Yıltan Bitirim

3. Asst. Prof. Dr. Mehtap Köse Ulukök

ABSTRACT

Database Schema Matching is a process which intakes multiple schema as an entry and yields back a mapping that classifies a similar component in these schemas. This process is mostly used to locate and identify semantically related-target. With this method or process it eases the finding and matching of divergent and randomly scattered data sets. It is one valuable tool for data processing and schema integration. Researches shows that various methods for schema matching based on different schema level matcher and classification criteria are proposed in order to find the most similar attributes and element in the schemas. Schema matching is classified into two approaches; Individual Match Approach and Combining Matchers Approach. In this thesis the individual match approach is used, which considers the schema level that is linguistic based. Past studies exhibited several methodologies to make the matching process in schema matching partially and fully automated, while in this thesis Convolutional Neural Network (CNN) methodology is proposed to implement an automated database schema matching engine with the aid of cosine similarity algorithm and Jaro Winkler algorithm. One of the powerful characteristics of the proposed methodology is that, it can be automated hence, less time is required to carry a particular task and more efficient if the task is more complex and if it is a larger scale task. The proposed methodology showed a very satisfactory result. The purpose of this thesis is to implement an automated database schema matching engine in addition to research and study the techniques and methodologies that is used for schema matching.

Keywords: Schema Matching, Individual Match, Multiple Matchers, Convolutional Neural Network.

ÖZ

Veritabanı Şeması eşleştirme, birden çok şemayı bir girdi olarak alan ve bu şemalarda benzer bir bileşeni sınıflandıran bir eşlemeyi üreten bir süreçtir. Bu işlem çoğunlukla anlamsal olarak ilişkili hedefi bulmak ve tanımlamak için kullanılır. Bu yöntem veya işlem sayesinde farklı ve rastgele dağılmış veri kümelerinin bulunması ve eşleştirilmesi kolaylaşır. Veri işleme ve şema birleştirilmesi için değerli bir araçtır. Araştırmalar, şemalarda en benzer özellikleri ve öğeleri bulmak için farklı seviyedeki şema eşleştiricisi ve sınıflandırma kriterlerine dayalı olarak şema eşleştirmesi için çeşitli yöntemlerin önerildiğini göstermektedir. Şema eşleştirme iki yaklaşım olarak sınıflandırılır; Bireysel Eşleşme Yaklaşımı ve Eşleştirici Birleşmesi Yaklaşımı. Bu tezde, dil temelli şema düzeyini dikkate alan bireysel eşleşme yaklaşımı kullanılmaktadır. Geçmiş çalışmalar şemada eşleştirme işlemini kısmen ve tamamen otomatik hale getirmek için çeşitli metodolojiler sergilerken, bu tezde Sarmallı Sinir Ağı (CNN) metodolojisi, kosinüs benzerlik algoritması ve Jaro winkler algoritması yardımıyla otomatik bir veritabanı şeması eşleştirme motorunun uygulanmasını önerir. Önerilen metodolojinin en önemli özelliklerinden biri, otomatikleştirilebileceğinden dolayı, belirli bir görevi yerine getirmek için daha az zamana ihtiyaç duyulması ve görev daha karmaşıksa ve daha büyük ölçekli bir görev ise daha verimli olmasıdır. Önerilen metodoloji oldukça tatmin edici sonuçlar göstermiştir. Bu tezin amacı, şema eşleştirme için kullanılan teknikleri ve metodolojileri araştırmaya ve incelemeye ek olarak otomatik bir veritabanı şema eşleştirme motorunu uygulamaktır.

Anahtar Kelimeler: Şema Eşleştirme, Bireysel Eşleme, Çoklu Eşleştiriciler, Sarmallı Sinir Ağı.

For my parent.

ACKNOWLEDGMENT

First and for most I would love to thank my family for the great opportunity and effort they invested in me. Without their support I wouldn’t have the courage to apply or even think of doing my Master’s degree.

A special thanks and full of love to my sincere Assoc. Prof. Dr. Duygu Çelik Ertuğrul, a member of the Eastern Mediterranean University, Department of Computer Engineering. Assoc. Prof. Dr. Duygu Çelik Ertuğrul had a really important role throughout my Master’s path. She would always help me with my doubts and even provide useful information and ideas without me asking for it. Without her guidance I wouldn’t have been able to achieve what I have achieved now. No matter how much I thanked her it wouldn’t be enough.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CNN	Convolutional Neural Network
CNs	Compound Nouns
CS	Complementary Schemata
DBMS	Database Management System
DDL	Data Definition Language
EM	Entity Matching
FEBRL	Freely Extensible Biomedical Record Linkage
GUI	Graphic User Interface
LC	Local Context
LD	Local abbreviation Dictionary
LSIM	Linguistic Similarity Coefficient
NORMS	Normalizer
OD	Online abbreviation Dictionary
ODL	Object Definition Language
OTA	OpenTravel Alliance Xml schema for travel industry
OWL	Web Ontology Language
RDF	Resource Description Framework
SEMINT	SEMantic INTegrator
SQL	Standardized Query Language
SSIM	Structural Similarity Coefficient
UML	Unified Modeling Language
WN	Word Net
WSIM	Weighted Similarity

Chapter 1

INTRODUCTION

Database Schema Matching is a process which intakes multiple schema as an entry and yields back a mapping that classifies a similar component in these schemas. According to Rahm & Bernstein survey's study, there are two main approaches for schema matching; Combining Matchers Approach and Individual Match Approach. Individual match approach is based on single matching criterion, which can be used either on schema level information or on instance data. In the schema level, the matching is performed either on individual element such as attributes, it can be either linguistic based or constraint based, or on combination of elements such as complex schema structure and it is constraint based. But in the instance data level the matching process is performed on element level and it can either be linguistic based or constraint based. While the combining matchers approaches are based on multiple matching criteria within an integrated hybrid matcher or by combining multiple match results that is obtained from different match algorithms within a composite matcher. In this thesis the individual match approach is used, which considers the schema level that is linguistic based [1].

The main objective of this thesis to develop an automated database schema matcher system and to research and study the techniques and methodologies that is used for schema matching. Moreover, to find a method that eases the findings and matching of divergent and randomly scattered data sets. Past studies exhibited several

methodologies to make the matching process in schema matching partially and fully automated, while in this thesis Convolutional Neural Network (CNN) methodology is proposed to implement an automated database schema matching engine with the aid of cosine similarity algorithm and Jaro Winkler algorithm.

The dataset that is used for this thesis is a data of four firms N11, LOGO, Serotonin and Mikro. The used dataset contains 302 field names and four tables. Each firm has one table and each table have a different number of fields and for each field name there is an expected field name; which is used for testing. The accuracy of the system that is implemented using the proposed methodology and this dataset is 84.5%.

This thesis is organized as follows. Chapter 2 presents a background about the techniques and methodologies that is used in this thesis. Chapter 3 covers the literature review of some related work. Chapter 4 explains the dataset that is used. The implantation of the system is presented and explained in Chapter 5. The experimental methodology is explained in Chapter 6. The system interfaces are presented in Chapter 7. The results of the experiment are demonstrated in Chapter 8. Conclusion is presented in Chapter 9.

Chapter 2

BACKGROUND

2.1 Schema Matching

Schema matching is one of the most common and important method to match between two or more schemas. Researches [1] shows that various methods for schema matching based on different schema level matcher and classification criteria are proposed in order to find the most similar attributes and element in the schemas. Schema matching is classified into two approaches; individual match approach and combining matchers approach. The following approaches does not distinguish between different types of schemas (relational, XML, object-oriented, etc.) and their internal representation, because algorithms depend mostly on the kind of information they exploit, not on its representation.

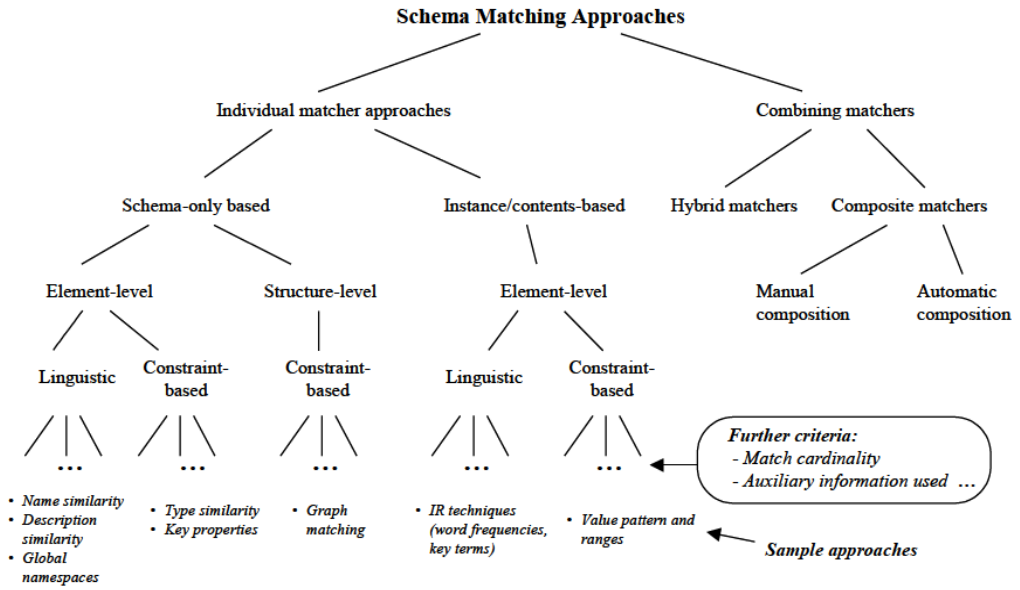

Figure 1: Schema Matching Approaches [1].

According to the previous research's schema matching was manually used with a graphic user interface which made it tedious, time consuming, error prone and thus expensive procedure, therefore it has a significant limitation. Recently schema matching had been modified and improved to be semi-automated and, in some cases, it can be fully automated but the effectiveness of the fully automated schema matching is unreliable hence it still needs the interference of the user or the programmer. Regardless of its setbacks it is still needed among relational datasets in databases, ontologies, XML and much more application. It is considered as one of the most important phases of data integration because most of the data integration applications requires matching between the schemas of the referred datasets. Different researches proposed different methodologies where each method that is proposed is based on different approaches, but it still served the same purpose which is matching two or more schemas. In this thesis a new method is proposed that is theoretically better than the previous methods because it have the ability to predict the patterns.

In next sections, the methodologies that are used in this thesis is explained. Separated sections considered in order to explain each methodology separately.

2.2 Convolutional Neural Network

CNN is basically a deep neural network, for supervised learning to analyze data. Usually in CNN for text it uses dense representation, it means that it can replace each word with dense vector, this is done using word embedding. At first CNN was invented for images and then later on it is shown that it is been effective in tasks such as word matching, sentence modeling, text classification, search query retrieval and sematic parsing. CNN is a very useful method for multiple tasks like classification, recognition, regression and prediction, it is used in images, speech and text. The best thing about

CNN is that it does not need feature extraction and its able to define the filter to be used without the human effort. It has the ability to predict the matching pattern at different layers in the semantics.

Since in this thesis is about words not sentences, character level CNN will be used. The different part in the character level CNN is that each character should be transformed into a vector and this is called character quantization. Each word will be converted into vector using the encoding function. Then the character level CNN structure will be built. A central role in analyzing the character level is that it does not need prior knowledge of the words, which makes it easier for CNN to adapt and develop irregular words caused by misspelling to different languages. The general CNN structure consists of three components:

1- One dimensional Convolutional layer: In this layer a dot product or a multiplication operation is done with a filter, where the filter can differ from one approach to another.
2- Pooling layer: There is two types of pooling; max pooling and average pooling. This layer reduces the quantity of the features and the calculations in the network, it chooses the maximum value of the parameter in the max pooling and it is the commonly used pooling, while in the average pooling it takes the average value.
3- Fully Connected Layer: This layer is the last layer in the CNN, it connects all the activations in the previous layers.

2.3 Cosine Similarity

Cosine Similarity is a function that is used to calculate the similarity between two variables such as strings or documents by converting the variables into vectors to

measure the cosine angle between the two non-zero vectors. Where the results of this function are between 1 and 0, where 1 indicates the two variables are exactly similar and 0 indicates that they are dissimilar, while the results that is between 1 and 0 indicates that its either intermediate similar or intermediate dissimilar it depends on the values.

$$\text{Cos}\,\theta \frac{A \cdot B}{\| A \| \| B \|} \quad (1)$$

2.4 Jaro Winkler

Jaro Winkler is a function that measures the similarity between two strings. According to Black's study [2] "Jaro measure is the weighted sum of percentage of matched characters from each file and transposed characters. Winkler increased this measure for matching initial characters, then rescaled it by a piecewise function, whose intervals and weights depend on the type of string" [2].

$$d_j = \begin{cases} 0 & if\ m = 0 \\ \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right) & otherwise \end{cases} \quad (2)$$

Chapter 3

LITERATURE REVIEW

3.1 Introduction

This chapter covers the literature review of some related works. Separated parts considered in order to explain and evaluate previous and related works in order to analyse the methods and find a method to match schemas using machine learning. Previous researches show that different algorithm and methods proposed in order to match the attributes and elements of the schemas.

3.2 Related Work

Sergey M. et al [3], presented a constant algorithm that can be implemented throughout many different situation and codes. This algorithm functions by capturing two different data forms; which can be schema or data structure and turning it into inputs and then processes it and later giving it out as an output that works as a scheme between the compared data set or structures. The compared data that is matched, a subdivision of the data is made relying on the targeted or desired outcome. In some cases, after the algorithm finishes the process human interaction is needed for some modifications and some proofreading of the data and it is known that, the “accuracy” of the system is calculated or measured by how many times a human interaction is needed per one algorithm run; as the number of interactions increases the accuracy decreases and vice versa. A small study is performed on the algorithm to check or evaluate, the accuracy scale of the algorithm versus the user.

The resemblances of the “Similarity Flooding” algorithm and its filters that had been introduced as an operative element in the administered testbed. The testbed is made or formed of a high-level algebraic process are mainly used for carrying out the models and the mappings with the help of scripts. The testbed assists the schema depiction, description and instance information that exists in RDF, XML, DDL [3].

Jayant M. et al [4], in this paper schema matching is reflected as a significant challenge or matter. A new algorithm called “Cupid” that is enhanced on past methods in many ways is discussed in this paper. However, it is thought that the algorithm is administered as an autonomous element. Cupid has some common global methodologies found in former algorithms. The methodology that is presented in this paper is known as a schema-based methodology but not an instance-based methodology. The example below summarizes the overall methodology.

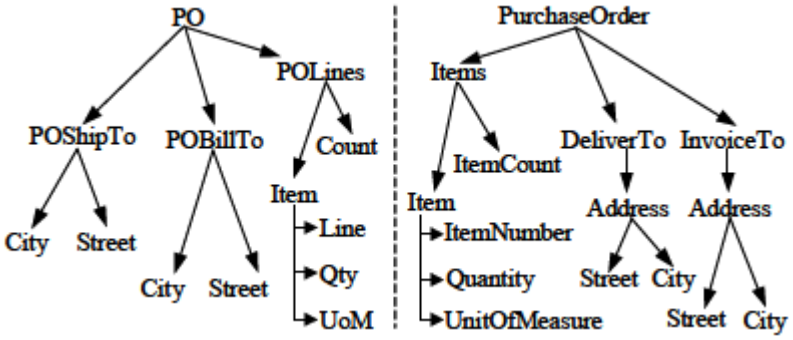

Figure 2: Purchase Order Schemas [3].

The aim is to match the two schemas of an XML format, “PO” and “Purchase Order” that is shown in Figure 1. A schema is converted into a graph that contains nodes that is characterized as schema component. As many schemas can be viewed as a similar

and simple data, even for an average viewer, but in reality, schemas can have a high degree of complexity, remodelling and alteration.

The attempt to solve the case, is by matching similar coefficients between two different set of schemas and later concluding the scheme from those coefficients that are matched. The coefficients, in the [0,1] scope, are then computed into two different stages. Stage one, which is designated as the "Linguistic Matching", which functions by joining and matching the components of each separate schema constructed by their names and data types. Thesaurus database is usually used to aid in name matching by distinguishing abbreviations ("Qty" for "Quantity"), acronyms ("UoM" for "UnitOfMeasure") and synonyms ("Bill" and "Invoice"). This develops an interposing pair of components, that can contain a coefficient of linguistic similarity.

The next stage or phase, depending on their correspondence of their adjacency and contexts the structural matching is carried out. The structural match partially relies on the linguistic matched components computed in stage one. This develops an interrupting pair of components, that can contain a coefficient of structural similarity. The weighted similarity (WSIM) is a mean of LSIM and SSIM.

$$\text{wsim} = w_{struct} \times \text{ssim} + (1\text{-}w_{struct}) \times \text{lsim} \quad (3)$$

constant w_{struct} is in the range 0 to 1. In the third stage or phase, known as "Mapping Generation", a scheme is yielded by only picking two of schema elements that has a maximum weighted similarity. Later a comparison is carried out by them to two others, to validate their application. This test the effectiveness of their methodology and it is a feasible model for future algorithm comparisons. Although it is assumed that some advancement had been made on the schema-matching problem, yet there is no exact statement confirming any clear solution for the issue. When machine learning is

applied to strategies, this combination of applications can later yield a firm and a very stable solution. Some strategies like, reusing well known matches by using pattern matching, natural language technology, instances, etc. The prolongated vision for Cupid is to make this algorithm inarguably having a global role in schema matching element that it is possible to be employed in all systems for schema integration, data migration, etc. The findings and product delivered is expected to be exceedingly extend path for this topic on hand [4].

Hanna K. et al [5], in this paper it has been observed that there is a trend toward researches involving “supervised (training-based)”, and “Hybrid methodology” to have a partially automated determining statement for a strategy known as “Entity Matching” (EM) for a particular task. Many deliberated structures involving the current structures like; Operator Trees and the Context Based Framework, operate superintended learners arises a numerical combination function or a match rules defining how several matchers ought to be joint for deducing a match decision. “Hybrid Framework” can deliver or provide substantial amount of methods for resolving setback likes entity matching function, specifically for obstructed or block and combined utilization of several matchers. Some of the regarded Hybrid Frameworks like, Freely Extensible Biomedical Record Linkage (FEBRL) which propositions a huge matching of many dissimilar functions that are blocking strategies, on the other hand the Context-based Frameworks, helps greatly when having a large number of learners and can also make context matching. Disturbingly, Hybrid Frameworks flexibility comes with some consequences such as the increased complexity of the user to decide which is the suitable method to be used regardless of employing the superintended machine learning approaches [5].

Jacob B. et al [6], characterized as a self-operating solution for the well-known issue of database schema matching. To get a perfect matched data between two semantically related schemas, Bayesian machine learning, statistical feature selection and the "Minimum Cost Maximum Flow Network" algorithms are used. An Attribute dictionary is formed, that contains a probabilistic knowledge that is achieved by the system from the examples that have been given by the domain expert. It is used to identify different data based on the probability that estimates the possible values of the identified data. The result of the probabilistic based method, that is used in matching every data of a schema with every data of another schema, is an individual score. For finding the best total match between the two schemas, an optimization process based on a "Minimum Cost Maximum Flow Network" algorithm with respect to the sum of the individual scores of the matched data are used. While for learning the efficient representation of the pattern a statistical feature selection method is applied by the "Automatch" for solving the very large dictionaries problem. The performance that is measured as the completeness of the matching process and the harmonic mean of the soundness exceeded 70%, which shows that the experimental result of the "Automatch" are optimistic. It is also stated that the attribute dictionary could be applied as a knowledge resource in other schema matching systems [6].

Wen-Syan L. et al [7], exhibited a methodology known as "Semantic Integration (SEMINT)", which is defined as the databases that are incorporated directly utilizing automated "catalog parser" to choose the metadata from the desired databases. Firstly, SEMINT exploits as specific application for picking and selecting the metadata from two different sources of databases. The application known as Database Management System (DBMS). Tags, markers or rather known as "Signatures" are formed by the metadata to depicts the features in each individual database. These feature signature or

markers are then utilized as a training data for neural network to concede those markers. The conditioned neural network can later distinguish the similar features based on the metadata or the signatures of the features in each database and indicates their resemblance. In this methodology, the metadata elimination is done by an automatic process; and how the matching of the similar features and the determination of their comparability is acquired by the system, is during the training process from metadata [7].

Serena S. et al [8], presented a method called “Normalizer of Schemata” (NORMS), an engine that act on schema tag normalization to increment the value of the matching tag that is selected from schemata. NORMS have many original features, mainly;

- NORMS enable the extend of abbreviations and improve the Word Net (WN) with Compound Nouns (CN) in automated manner.
- NORM can supply the GUI that aids the user during the normalization procedure and enable them to modify and improve the automated outcomes generated by fixing and correcting the potential errors.
- NORMS can also allocate the potency of automating an annotation of the schema components with respect to WN.
- Finally, the output can be achieved by multiple matching operations.

The methodology is constructed of four core segments: Wrapper, Schema Label Normalizer, Lexical Annotator and an Output Generator [8].

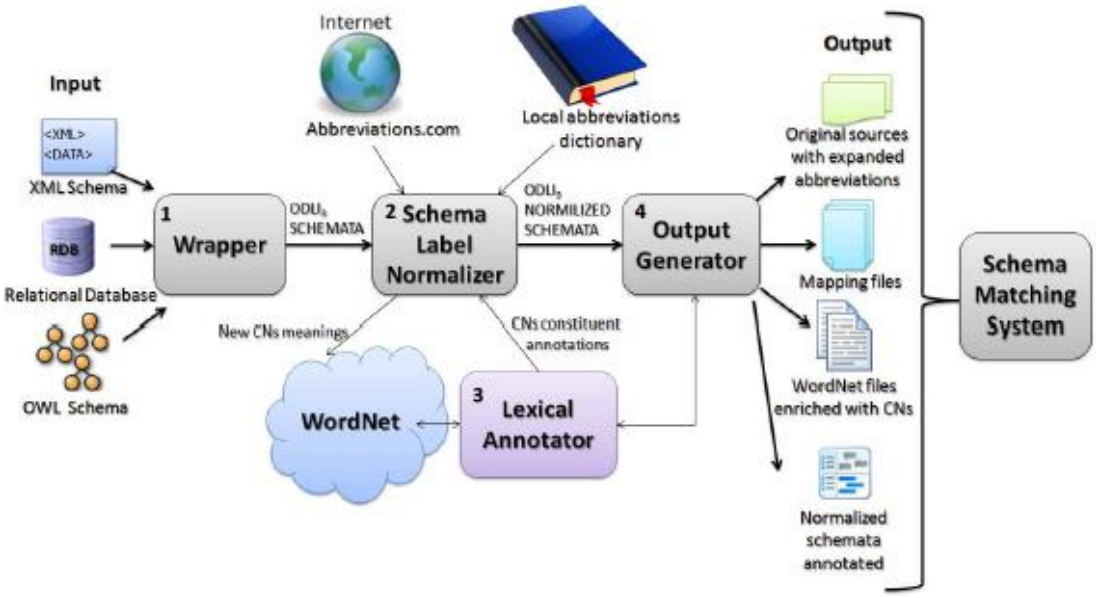

Figure 3: Structure of NORMS [8].

Hong-Hai D. et al [9], developed the “Combining Match Algorithm (COMA)” as a framework to merge several matchers in an adjustable way. The match is an operation that takes two schemas as input and chooses a mapping specifying which elements of the input schemas logically correlate to each other. The result that is obtained from the match is a set of mapping elements determining the matching schema elements together with a similarity value between 0, which indicates that there is no similarity, and 1 indicates that there is a strong similarity, demonstrating the validity of their consistency.

Subsequent to the transformation of the inner graph format, the corresponding algorithms compute the similar values for the schema that are traversed to establish all the schema components. The schema is representing the components of its sequences as nodes, subsequently restricting any links from the origin of the schema to the similar nodes.

User feedback application in COMA helps the user to obtain the matched and mismatched data that is only comprised of validated matches from the former matching replications. This step aids in ensuring that the matches and the mismatches are confirmed and then appointed to a maximum and minimum resemblance values. These values are stored as constants so that, in future entries or application, these values won't be affected by another matcher amidst other matcher implementation step. The similarity values obtained from the previous step, is later supplied to user in order for them to decide the similarity computations of the neighbouring of the respective components, hence resulting in many improvements in the matching precision and accuracy of the structural matchers [9].

Table 1: Summary of Algorithms.

Author name and ref. no.	Algorithms	Definition	Data used
Sergey M. et al [3]	Similarity Flooding	It is an easy algorithm that is used to match various data structure and it is based on fixpoint calculation.	Graphs
Jayant M. et al [4]	Cupid	It is a schema-based algorithm that is enhanced on past methods in many ways.	Schema tree and XML Schema
Hanna K. et al [5]	Entity Matching	It is a framework that arises a numerical combination function or a match rules defining how several matchers ought to be joint for deducing a match decision.	XML and Databases
Jacob B. et al [6]	Automatch	It is based on a knowledge base of schema attributes which is created from examples.	Attribute Dictionary
Wen-Syan L. et al [7]	SEMINT	It is an instance-based matcher that identifies the attributes in two schemas with the match signatures.	Databases
Serena S. et al [8]	NORMS	It is a discrete implementation that makes the schema label normalized.	Database, XML and OWL. Word net.
Hong-Hai D. et al [9]	COMA	It is an overall match system that support various applications and several schema types.	Directed acyclic graph.

Chapter 4

DATABASE USED

4.1 Introduction

The data that is used in this thesis are four databases of four e-commerce firms, which are N11, Logo, Mikro and Serotonin. These databases will be discussed and explained in details in this chapter, each firm will be discussed in a separate section.

4.2 N11

N11 [10] is an e-commerce database which consist of several tables; Orders List table, Category table, Cities table, Shipment table, Product table, Products Service table and etc., from this database tables product table is used in this thesis. The product table consist of seven columns; field name, field description 1, field description 2, data type, length, mandatory and key. This table contains the firm's product details such as product name, product type, product order and etc.

Table 2: N11 Product Table.

Table field	Field description 1	Field description 2	Data type	Length	Mandatory	Key
SellerCode	Product Code		varchar	25	1	FK
title	Product Name		varchar	35	1	
price	Product Base Price		double	8	1	
currencyType	Product list price currency		varchar	30	1	
url	Product official URL		varchar	30	1	

4.3 Logo

Logo database is an e-commerce firm [11], which consists several tables that contains the products details, payment transactions, shipment details, factory parameters and etc. The table that is used in this thesis from this database is the items table which contains the products details. The columns of this table are field names, field description 1, field description 2, data type, length, mandatory and key.

Table 3: Logo Items Table.

Table field	Field description 1	Field description 2	Data type	Length	Mandatory	Key
item_code	Material code		Varchar	25	1	Pk
item_name	Material description		Varchar	51	0	
item_price	Price		Double	8	0	
vat	Tax		Double	8	0	
photo	Picture		Byte	1	0	

4.4Mikro

Mikro is an e-commerce firm [12], its database consists of all the details of the firm, accounting table, brand table, order table, addresses table, stocks table, staff table and etc. Stocks table is used from this database which consist the following columns; filed name, field description 1, field description 2, data type, length, mandatory and key.

Table 4: Mikro Stocks Table.

Table field	Field description 1	Field description 2	Data type	Length	Mandatory	Key
stock_code	Product code		Varchar	25	1	Pk
stock_name	Product name		Varchar	50		
stock_retailtax	Retail tax rate		Tinyint			
stock_currencytype	Currency id		Tinyint			

4.5 Serotonin

Serotonin [13] is an e-commerce firm and their database contain several tables and the table that will be used in this thesis is the product table which contains all the product details, it consists of the following columns; filed name, data type, length, mandatory and key.

Table 5: Serotonin Products Table.

Table field	Data type	Length	Mandatory	Key
Product_code	Varchar	25	0	PK
Product_name	Varchar	25	1	
sales_price	Varchar	20	0	
tax	Float	18	0	
currencytype	Varchar	5	0	

Chapter 5

SYSTEM PROPOSED

5.1 Introduction

The proposed system consists of five modules, pre-processing module, search module, database, matching module and CNN classifier module. These modules will be discussed and explained details next. In Section 5.2 system controller will be discussed, in Section 5.3 pre-processing will be explained step by step, in Section 5.4 search module will be explained, in Section 5.5 database will be discussed, in Section 5.6 matching module will be explained and in Section 5.7 CNN classifier will be explained.

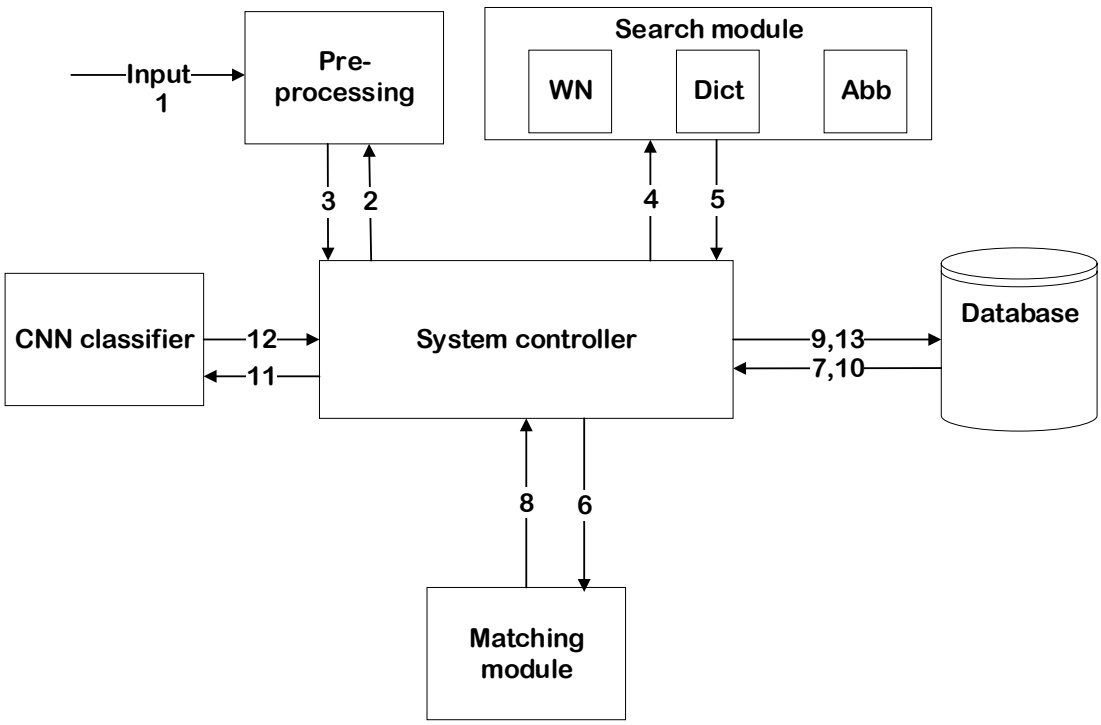

Figure 4: General Architecture of the System.

5.2 System Controller

System controller controls the input and output of the data from each module, the steps of the controller mechanism are:

1. The system controller first calls the pre-processing module to pre-process the input and returns the data after its been pre-processed to the controller.
2. Next the controller calls the search module to check if the entry is in abbreviated form or not by checking the abbreviation file. Moreover, to check if the entry is spelled correct or not by checking the dictionary file and to get the synonym of the entry, then returns to the system controller.
3. If the database is not empty the system controller calls the matching module and match the data that is in the database with the entered data and returns data to the database.
4. Otherwise the system controller saves the data to the database directly.
5. Next the system controller extracts the data from the database and sends it to the CNN classifier.
6. Then the result of the CNN classifier is sent to the system controller to save it in the database.

5.3 Pre-processing Module

To start off, data needs to be understandable and to make it understandable, it should be preprocessed. In pre-processing, removing the prefixes and underscores is done by using the “Split” functions.

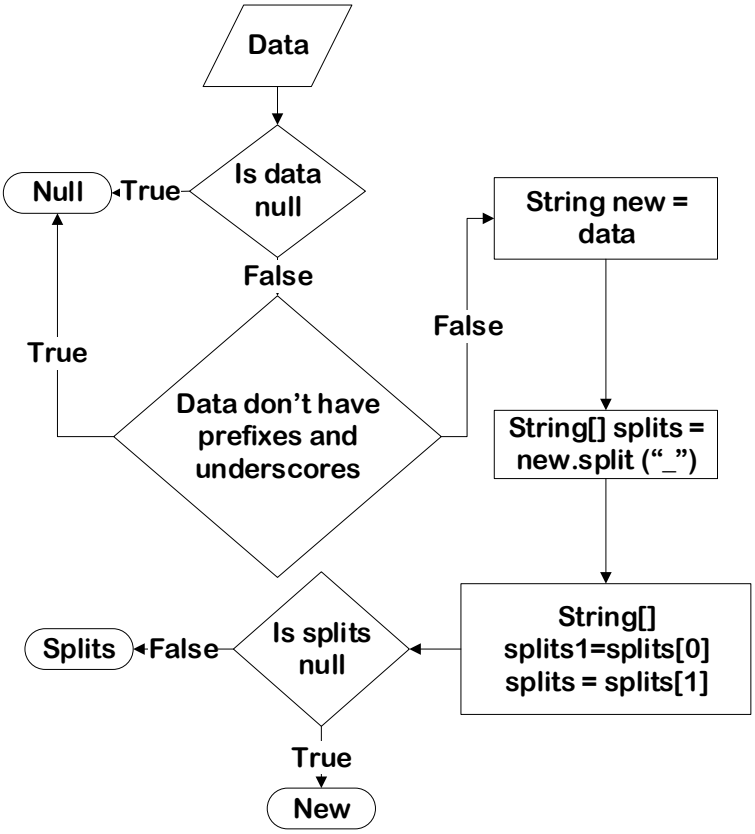

Figure 5: Pre-processing Flowchart.

The final step in the preprocessing is converting the data into lower case letters by using a ready function in java called "toLowerCase" and for removing the spaces "replaceAll" function is used. The pre-processing process is an essential process for the system to exhibit a meaningful and understandable outcome, without this process the system would show off meaningless and un-understandable data or outcome.

5.4 Search Module

In this module there is three tasks, search for the full form of the abbreviated data from the abbreviation file, if the entry is in the abbreviation form.

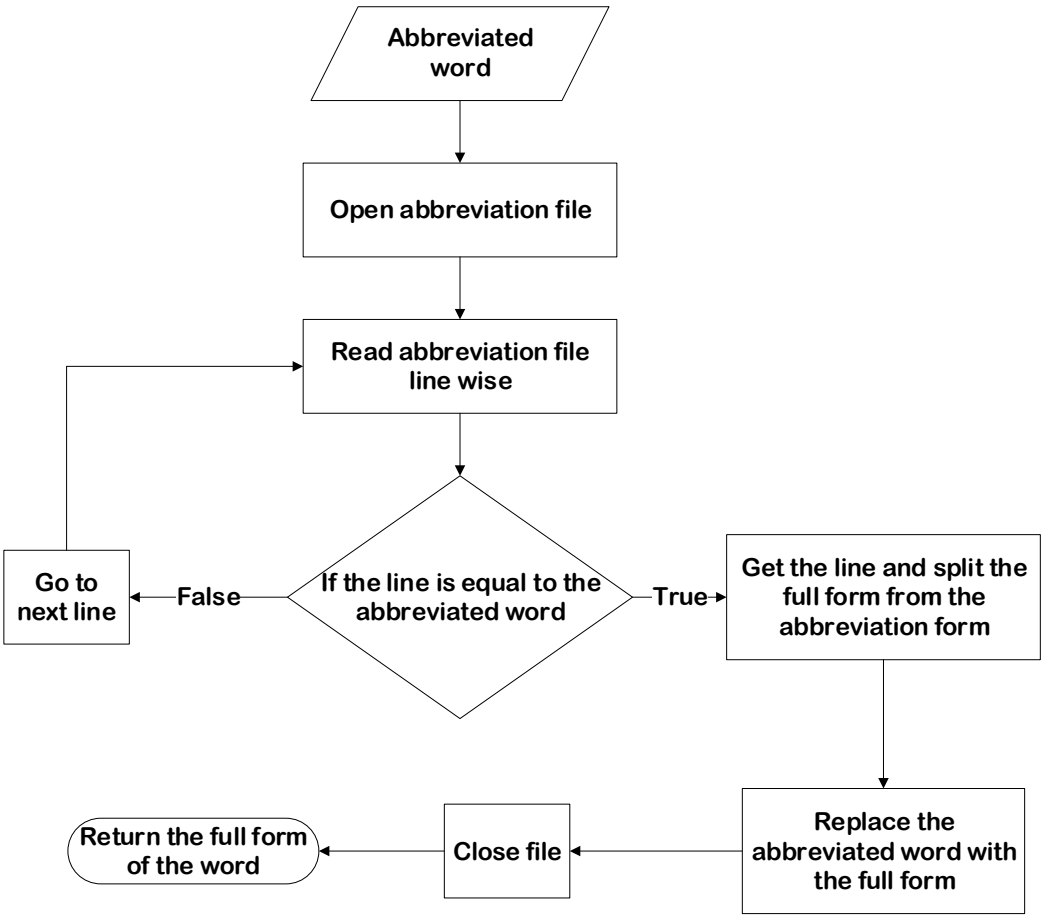

Figure 6: Flowchart of Replacing Abbreviated Word with the Full Form.

Then checks if the spelling of the entry is correct or not by checking the dictionary file. The algorithm that is implemented for checking the spelling of the entered data using a ready function in java called “SpellChecker” and an outsource dictionary, this function is very useful, since it have properties such as linking a dictionary to the function, a listener to the data, and check spelling to check the spelling of the data from the dictionary. The check spelling property is in the integer form it returns -1 which means the spelling is correct and otherwise for wrong spelling, for this reason word tokenizer is used to token the data. If the result of this function is -1 then the spelling of the entered data is correct but if gives otherwise, the result is sent to another function called “Misspelling” function, this function uses a Jaro Winkler formula to get the correct spelling of the data.

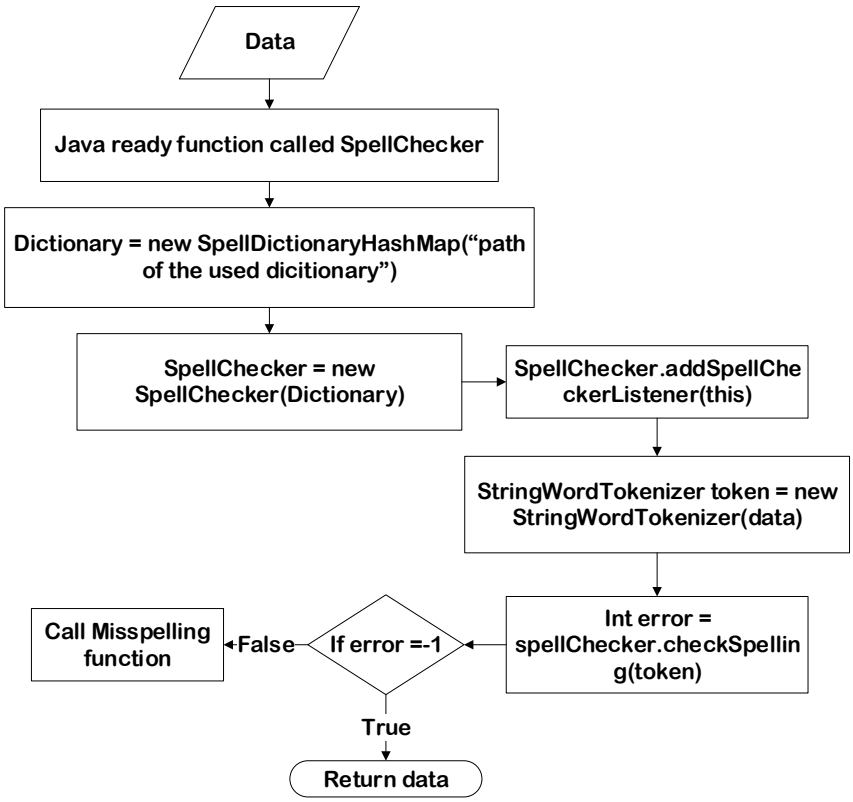

Figure 7: Flowchart for Spelling Check.

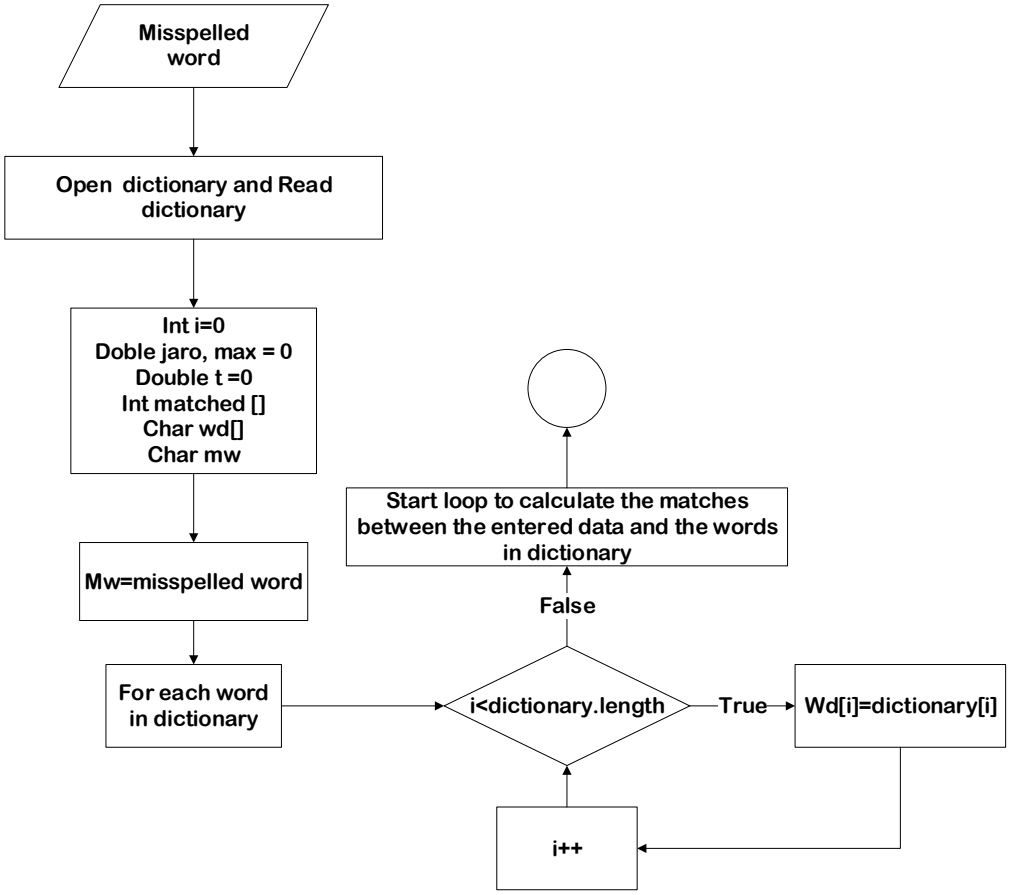

Figure 8: Flowchart of the Misspelling Function.

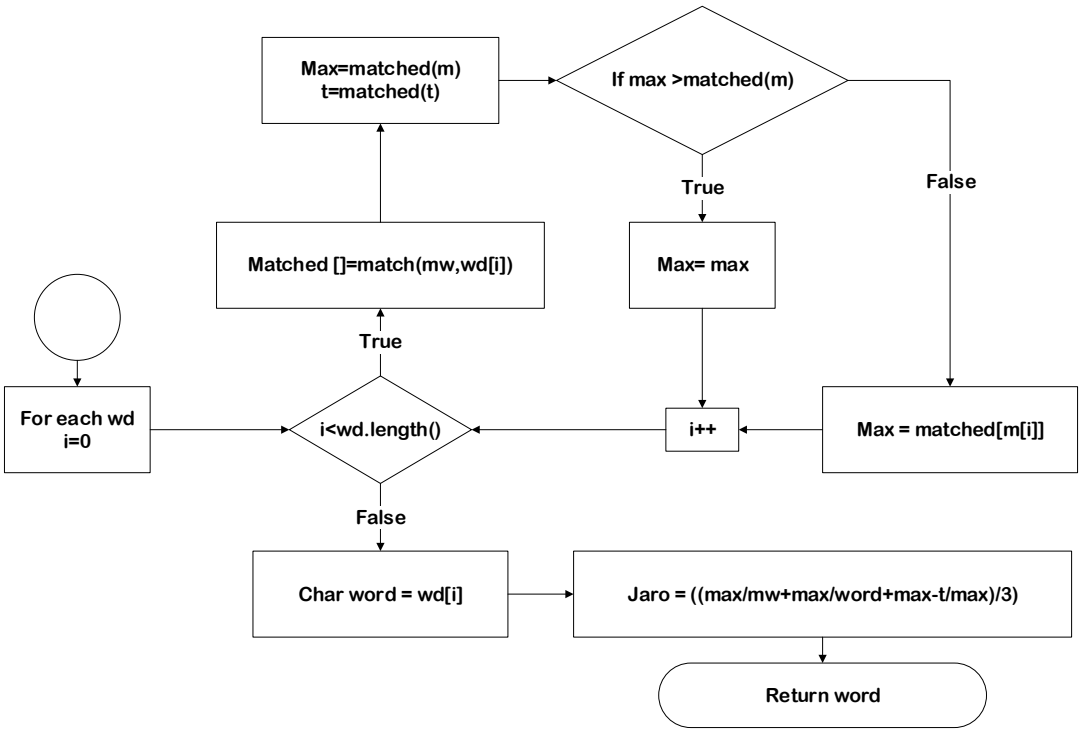

Figure 9: Flowchart of the Loop for Misspelling Function.

The match function is the function that is called by the misspelling function, this function is used to match the letters of the two data that is entered. The result of this function is the number of the matched letters and the transposition value. The transposition value is the value of the times that mismatch occurred. These results are used to calculate the Jaro Winkler distance. The last task in the search module is searching for the synonym of the entry in the word net.

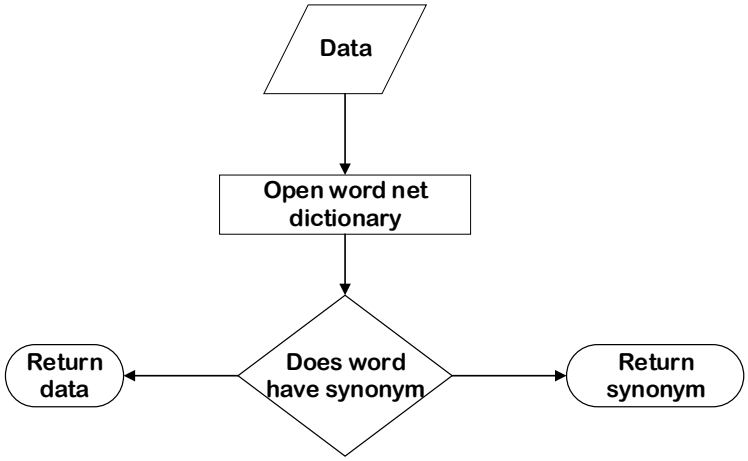

Figure 10: Flowchart of Getting the Synonyms.

5.5 Database

The database consists of five tables, the four tables are the tables from the four e-commerce databases. These tables are the training data and it is entered by the user. The fifth table is the table that contains the common field names of the four tables that is obtained by the proposed system.

5.6 Matching Module

In this module the matching process is done by checking if the entered data or the synonym of the entered data is available in the database.

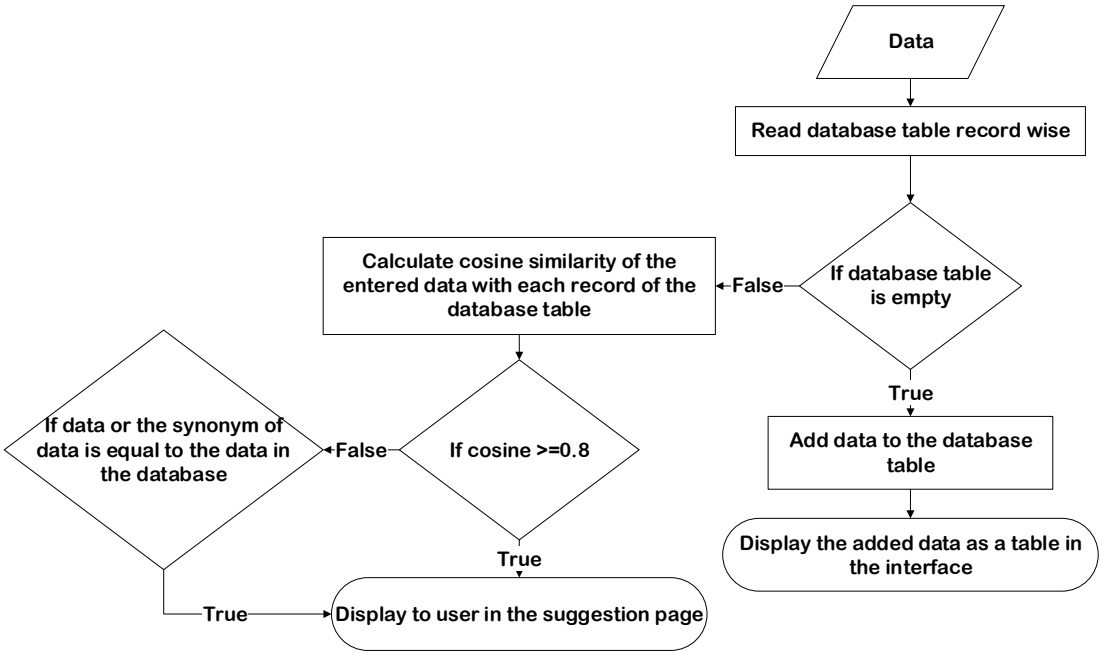

Figure 11: Matching Process Flowchart.

5.7 CNN Classifier Module

This module matches all the data in the database and save it in one table named "Schema". Moreover, this table is constructed when the match process is done using the CNN classifier, where it groups similar data in one group and then it saves the matched data without repetitions in the "Schema" table.

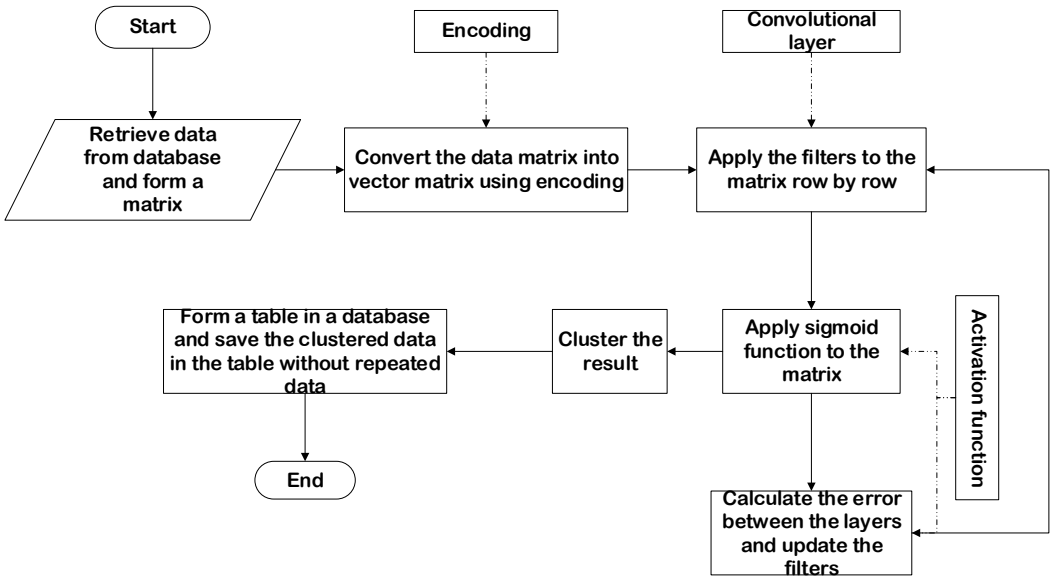

Figure 12: CNN Classifier Flowchart

The cluster process in this algorithm works in a simple way. It saves the first value that is obtained from the activation function and save it in a list and the index of this list will be one, and for the next value it checks if the available list contains a similar value, then the value will be saved in this list, but if the value is not similar to the value that is in the list then it will create a new list and save the value there. This process is done for all the values that will be obtained from the activation function; these lists are the clusters. Moreover, the encoding process is a function that converts the data into vectors using the ascii values.

Chapter 6

EXPERIMENTAL METHODOLOGY

6.1 Introduction

This section contains the thesis methodology that describes the research approaches, methodology structure, the structure of the schema matching and finally Research Limitations.

6.2 Dataset Description

The dataset that is used for this thesis is a data from four firms N11, LOGO, Serotonin and Mikro. This dataset contains 302 field names and four tables. From each firm one table is used, which contains the products details. Moreover, each table have a different number of fields and for each field name there is an expected field name; which is used for testing. Therefore, the dataset is divided into two sections; the table name and field name section are used for training while the expected field name is used for testing.

6.3 Research Approaches

For the purposes of this thesis the research approach followed is instructive. For this method, researchers used CNN for text classification and prediction. From the previous articles I was inspired to implement the CNN algorithm for the schema matching. The 302 field names are used in this thesis to test the proposed algorithms, each field name has been converted to vectors by the encoding function and formed a matrix to be processed for the next step. The results of the algorithms have been compared with the expected field names that is available in the dataset.

6.4 Methodology Structure

As mentioned earlier the general CNN structure consist of three main components, but in this thesis a character level CNN is implemented and the structure of this method is not very different from the general method structure. In this structure there is three layers. The first layer is the convolutional layer, in this layer the vectors is multiplied with filters using the dot product operation, since it is 1-D convolutional the dot product is done row by row, and the results is summed up to get one result for each row. The filter that is used in the convolutional layer is chosen by taking the longest field names. The next layer is the activation function, in this layer, the result is entered to the activation function which is the sigmoid function, it squashes the result of the previous layer to a range between 0 and 1. The sigmoid formula is:

$$sigmoid(y) = \frac{1}{e^{-x}} \tag{5}$$

where -x is the value obtained from the convolutional layer. In the last layer the algorithm gets the final result and the error with sample output. The final result that is obtained by choosing the maximum value that is nearest to the threshold that is defined. Since this algorithm is a back propagation, it will calculate the difference in each layer and subtract these error values from existing weight values, to update the filter and threshold.

6.5 Schema Matching Structure

In this thesis, database schema matching using character level CNN with sigmoid and back propagation clustering will be discussed and explained. This combination of processes is used together to make a system that retrieve the matched data and create a new table that contain the newly matched data into the database.

When the system first executes, a connection to the database is established. Then the user is able to enter the data to the system through the user interface, then the system preprocesses the entry and adds the data to the database. While the user adding the data, the system checks if there is any similar data entered to the table. If there are any similarities, the system suggests for the user the available data and the user can choose whether to take the suggested data or not. This matching process is done by checking the database whether the entry already exists in the table assigned by the user. The matching process in this step is only in the table scope.

Moreover, the matching process in the database scope is done using the CNN clustering. At this point the data is fully entered to the database and the training matrix is extracted from the data entered. The training matrix is entered to the convolutional layer so that the filter is applied to the matrix and form a new matrix. Then this matrix is entered to the activation function to suppress the results to an array of a range between 0 and 1. An array with several values for each word is obtained and fed to the next layer, by which the layer compares and locate the maximum value which is nearest to the threshold that is defined within the array given. i.e. the first maximum value chosen is saved in a group corresponding to its position and it is done for the next array, if the maximum value of the next array has a group with the similar values then it is relocated to that group otherwise it is saved to a new group. The maximum value is later relocated to a group containing similar maximum values obtained from other arrays. Since this algorithm is a back propagation, it will calculate the difference in each layer and subtract these error values from existing weight values, to update the filter and threshold. Finally, as a result, the algorithm groups together all similar words and save it into the database as a new set of data in a new, separate table without any repetitions.

On the other hand, if the entered value gives invalid results, CNN trains the weight values so that it can give a valid value.

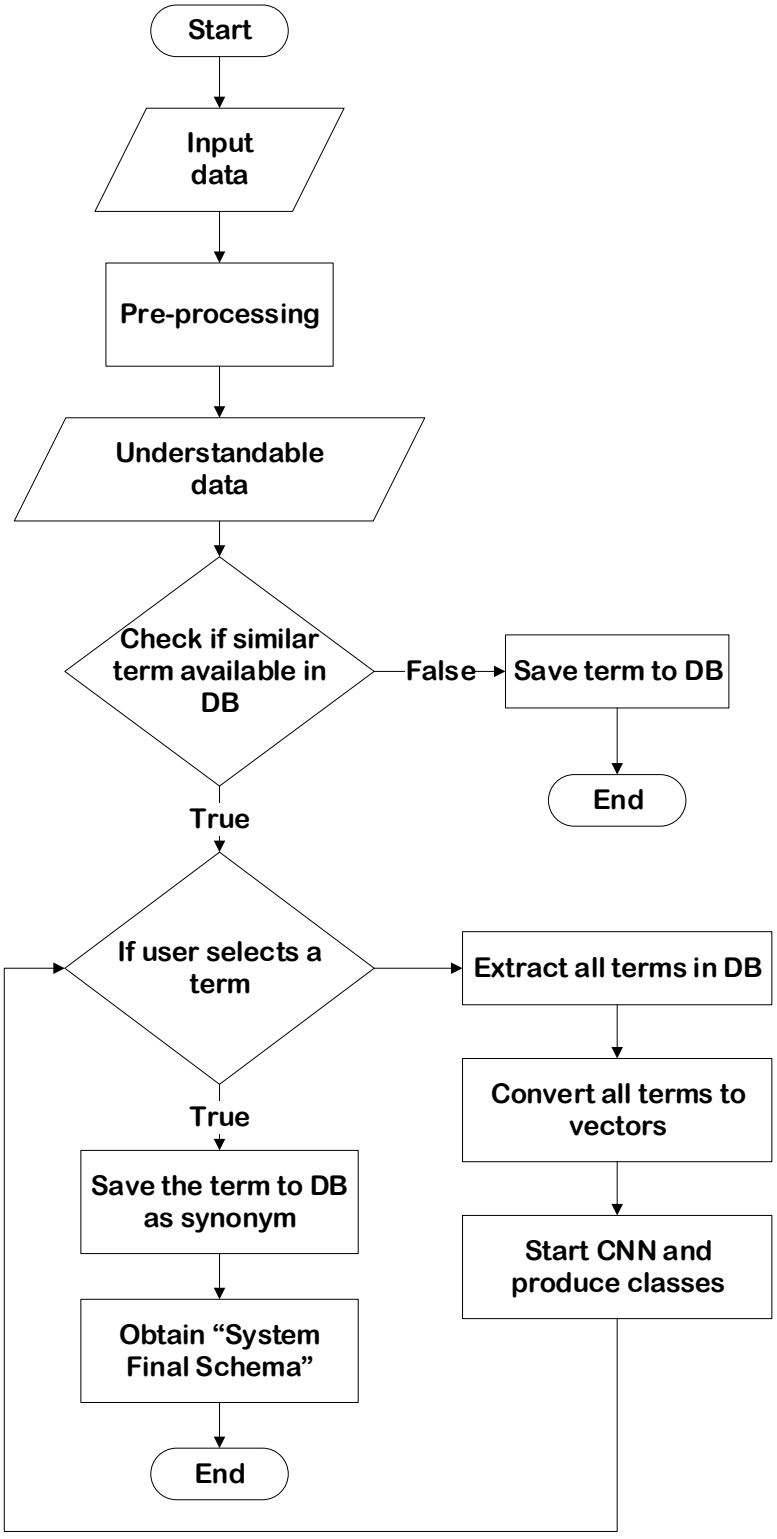

Figure 13: Schema Matching Structure.

Chapter 7

SYSTEM INTERFACES

7.1 Introduction

The proposed system is implemented in Java using NetBeans IDE 8.2 and SQL Lite studio. In this section the interface of the system will be discussed and how the user enters the data will be explained.

7.2 Main Interface

The main interface allows the user to enter the data by filling the following text boxes; in the table name text box the firm name followed by the table name is entered, by putting an underscore between the firm name and the table name. The field name is entered in the field name text box. Next, data type of the field name is chosen from the combo box. Subsequently, the length of the chosen data type is entered in the length text box. If there is a default value for the entered field name it is entered in the default value text box. If the entered field name is mandatory then the mandatory text box is filled with the value 1 and if it is not mandatory then it is filled with the value 0. The key type of the entered field name is chosen from the key combo box (primary key, foreign key, none). If there is a data description for the entered field name then it is filled in the data descriptions text boxes as shown below.

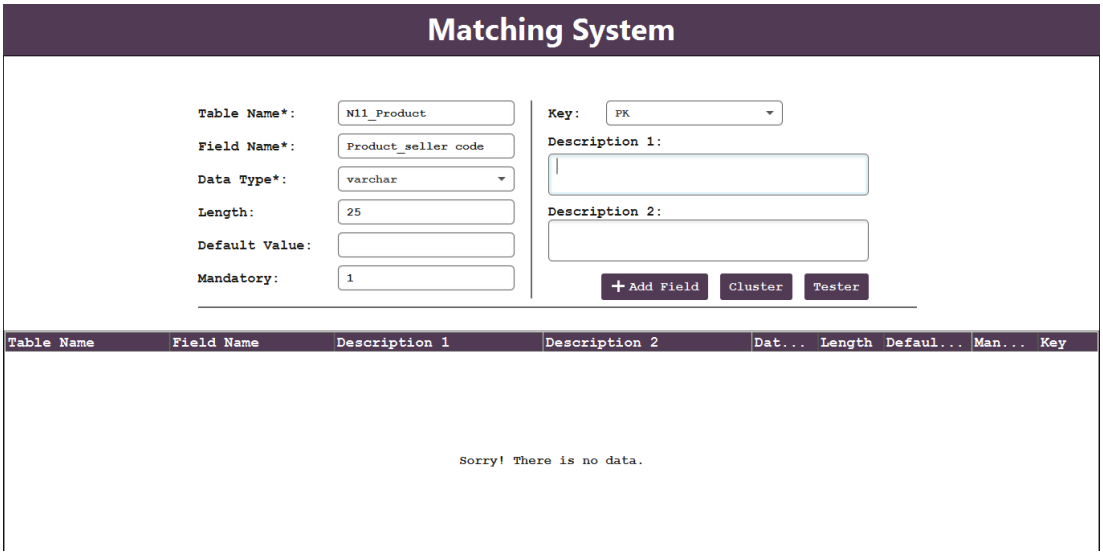

Figure 14: Main interface

When the add field button is clicked, the data will be entered to the system database with “Exact Match” relationship. The data that is entered will be displayed for the user in the interface as shown below.

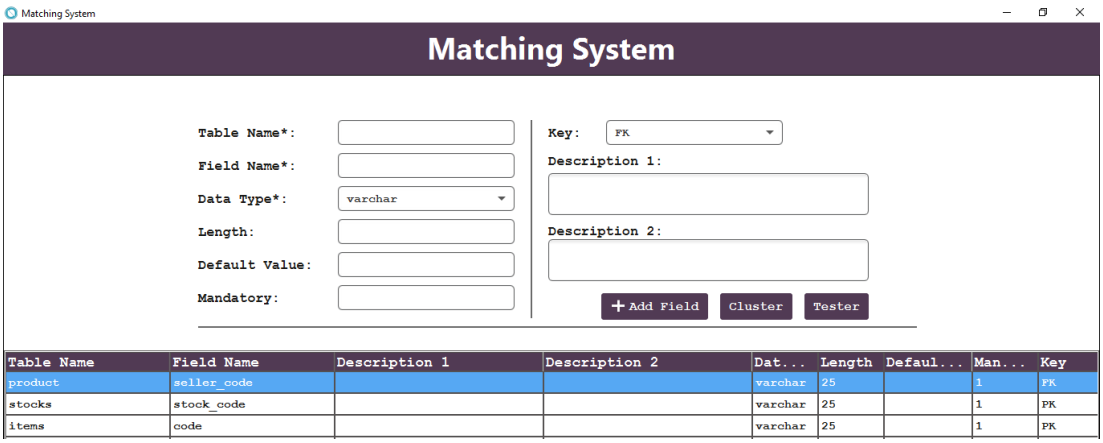

Figure 15: Interface showing entered data

Moreover, when the user enters data that already exist in the database it shows the user the available data while the user is typing it, as shown below.

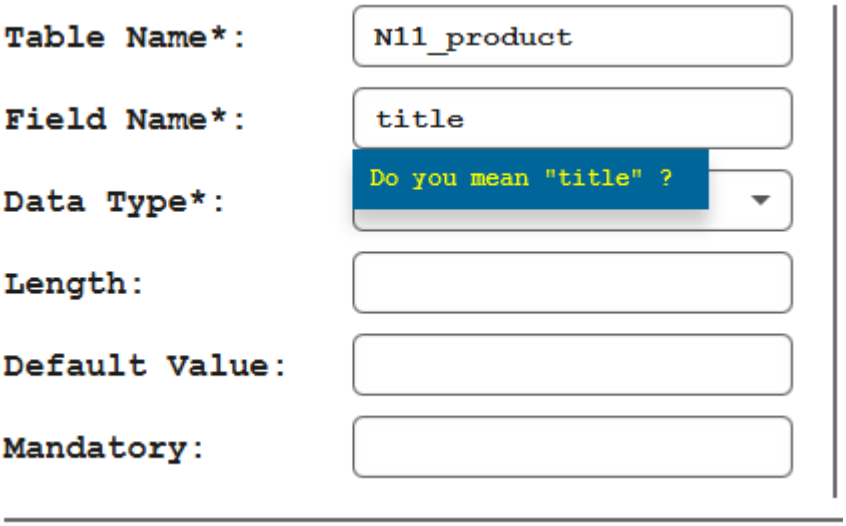

Figure 16: Asking the user if this is the data meant

7.3 Suggestions Interface

The suggestions interface is displayed when the add field button is clicked and there is similar data of the entered data in the database.

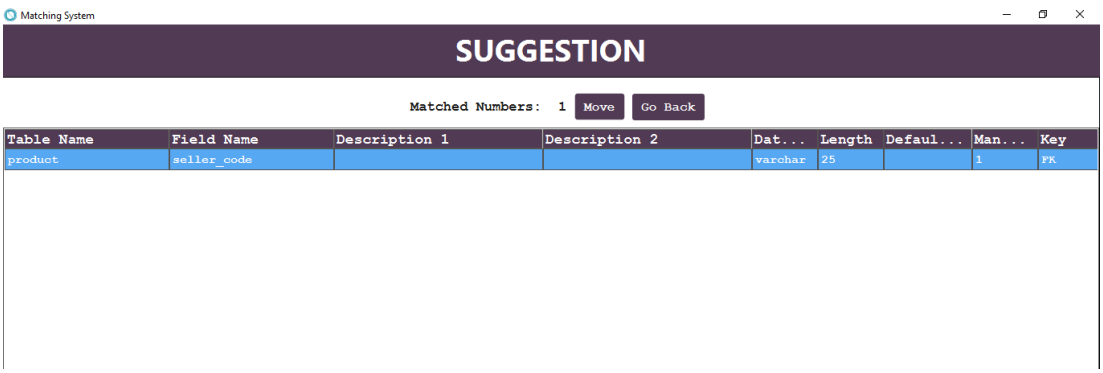

Figure 17: Suggestion interface

If a data is chosen from this interface then it will be added to the interface's table and to the database as a synonym by clicking the move button. However, if none of the suggested data is chosen, then the main interface is displayed and the entered data will be saved to the database by clicking go back button.

7.4 Clustering Interface

Clustering interface is appeared when the cluster button that is in the main interface is clicked. The clustering interface displays the data that is grouped in one cluster according to its similarity.

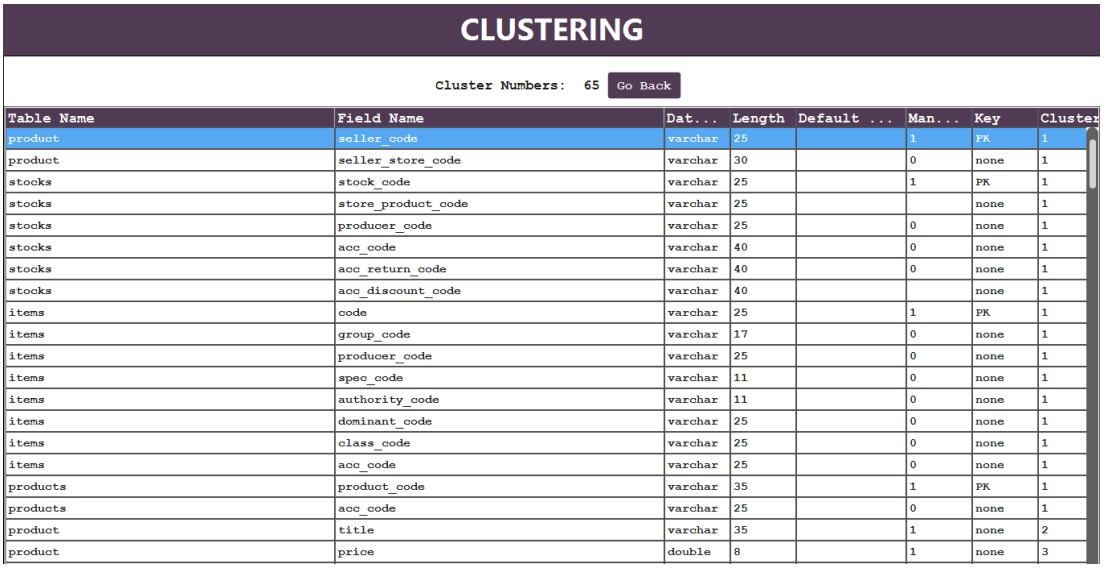

Table Name	Field Name	Dat...	Length	Default ...	Man...	Key	Cluster
product	seller_code	varchar	25		1	FK	1
product	seller_store_code	varchar	30		0	none	1
stocks	stock_code	varchar	25		1	PK	1
stocks	store_product_code	varchar	25			none	1
stocks	producer_code	varchar	25		0	none	1
stocks	acc_code	varchar	40		0	none	1
stocks	acc_return_code	varchar	40		0	none	1
stocks	acc_discount_code	varchar	40			none	1
items	code	varchar	25		1	PK	1
items	group_code	varchar	17		0	none	1
items	producer_code	varchar	25		0	none	1
items	spec_code	varchar	11		0	none	1
items	authority_code	varchar	11		0	none	1
items	dominant_code	varchar	25		0	none	1
items	class_code	varchar	25		0	none	1
items	acc_code	varchar	25		0	none	1
products	product_code	varchar	35		1	PK	1
products	acc_code	varchar	25		0	none	1
product	title	varchar	35		1	none	2
product	price	double	8		1	none	3

Figure 18: Clustering Interface

Chapter 8

EXPERIMENTAL FINDINGS

8.1 Introduction

In this section, the results and evaluation of the proposed methodology used in this thesis will be explained. The tables that is used are from four different databases (N11, LOGO, MIKRO, SEROTONIN). A single table is used from each database, “N11_product” table consist of 25 field names, “LOGO_items” table consist of 64 field names, “MIKRO_stocks” table consist of 198 field names and “SEROTONIN_products” table consist of 15 field names. The total field names are 302 field names. Moreover, an “Expert Schema” is constructed manually from these four tables and “System Final Schema” is obtained by the system. For evaluating the results True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) matrix are used to calculate Precision, Recall, F1-score and Accuracy. There are two types of evaluation, the first evaluation is the database tables versus the user input tables. The second evaluation is the “System Final Schema” versus the “Expert Schema”. Full explanation of the results is stated next.

8.2 Evaluation 1: Database Tables Versus User Input Tables

To calculate the Accuracy, Precision, Recall and F1-score, confusion metric (TP, TN, FP and FN) values must be obtained. To obtain these values, a comparison between each user input table and database tables is done.

TP is the exact match or the synonym of the input that is predicted by the system and suggested to user. TN is the wrong prediction of the input that is predicted by the system and suggested to the user. FP is the wrong terms that the system could not predict of the input. FN is the correct terms that the system could not find of the input and not suggested.

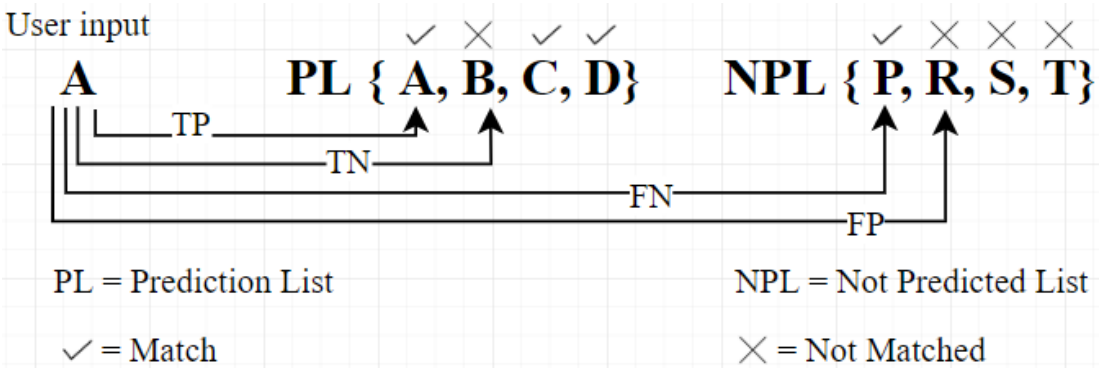

Figure 19: TP, TN, FP and FN example.

For instance, the user table involves the terms {product code, name, sales price, tax, currency type, photo name, order, state, discount price, stock piece, statement, group id, expiration date}. The first term “product code” seeks to be matched by the system terms that is available in the system database. Assume that the term “product code” is matched with the predicted list (PL) {product code, stock code, product number, item code, item id, stock number, etc.} and correctly found by the system. Second assumption is that the wrong predicted list (NPL) for the “product code” are {id, product name, stocks, number, stock name, etc.}.

Assuming that the input is “product code” and the system suggests the synonym or the exact term as “product code”, “stock code”, “item code” and etc., then this is considered as TP. If the system suggests a term that is incorrect term for the input as “code”, “id” and etc. then this is considered as TN, but if the system did not suggests

any wrong terms for the input then this is considered as FP, and if the system did not suggest any correct terms for the input then this is considered as FN.

Table 6: Confusion Matrix 1

		Actual	
		True (T)	False (F)
Predicted	Positive (P)	190	30
	Negative (N)	63	19

Confusion matrix is useful for calculating the accuracy, precision, recall and f1-score. Where accuracy calculates how accurate is the algorithm.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

Precision refers to the percentage of the results that is relevant to the actual data.

$$precision = \frac{TP}{TP + FP} \tag{7}$$

Recall calculates the percentage of the total results that is correctly matched with the actual data.

$$recall = \frac{TP}{TP + FN} \tag{8}$$

F1-score calculates the accuracy of the test, both the precision and recall are used to compute the F1-score.

$$f1 - score = 2 \times \frac{recall \times precision}{recall + precision} \tag{9}$$

After the results are obtained from the confusion matrix, the values for the accuracy, precision, recall and F1-score is computed and recorded on the following diagrams.

Table 7: Testing Results 1

Accuracy	0.840
Precision	0.864
Recall	0.910
F1-score	0.886

8.3 Evaluation 2: System Final Schema Versus Expert Schema

The result of the system is a final schema of the input tables of companies. The schema is saved to the system database to use next matching processes. The result schema is titled as "System Final Schema", is compared to the "Expert Schema" that is manually constructed. The field names that is in the "System Final Schema" is decreased to 254 field names from 302 field names and the field names that is in the "Expert Schema" is decreased to 239. This is because each field name is stored once without its synonym. The confusion metric (TP, TN, FP and FN) values are calculated after comparison of the schemas.

Table 8: Confusion Matrix 2

		Actual	
		True (T)	False (F)
Predicted	Positive (P)	150	24
	Negative (N)	52	13

Table 9: Testing Results 2

Accuracy	0.845
Precision	0.862
Recall	0. 920
F1-score	0.890

8.4 Results Comparison

In this section, the findings computed from the schema matching using CNN are compared with two different articles relevant to the same subject in hand. Each article

had a different method but used for schema matching. A comparison for each article is explained next.

8.4.1 Comparison with Schema Matching Using Machine Learning

According to Sahay T. et al, “Schema Matching Using Machine Learning” [14], two methods are used for clustering and linguistic matching. The first method that is used is the centroid method and the second method is the combined method. In the centroid method two types of clustering methods are used to group similar attributes of the source schema together, the clustering methods that are used are Kohonen Self Organising map and K-Means Clustering [14].

The results of these two methods are compared with the results that is obtained from this thesis and is shown below.

Table 10: Results of Comparison between Our Work and Sahay T. et al [14] Centroid Method.

	Our work	**Sahay T. et al [14] Centroid Method**
Precision	86.4%	52.70%
Recall	91%	100%
F1-score	88.6%	69%

As shown above, the performance results of the methods proposed in this thesis are better than the experimental results of the study [14]. This is due to the usage of the word net dictionary, which finds and eliminates synonym words for each field name entered. Moreover, the CNN algorithm updates the weight of the entered field name in each iteration which gives better results.

Table 11: Results of Comparison between Our Work and Sahay T. et al [14] Combined Method.

	Our work	Sahay T. et al [14] Combined Method
Precision	86.4%	54.00%
Recall	91%	100%
F1-score	88.6%	71%

Even though the combined method performs better than the centroid method, still the proposed methods performs better than these two methods.

8.4.2 Comparison with an Instance Based Approach

Mehdi O. et al, "An Approach for Instance Based Schema Matching with Google Similarity and Regular Expression" [15], in this paper an approach for schema matching is proposed which is instance based. This approach relies on the combination of google as a web semantic and regular expression [15].

A comparison between the results that is obtained using this approach and the results that is obtained from this thesis is shown below.

Table 12: Comparison Results between Our Work and Mehdi O. et al [15] Instance-Based Approach.

	Our work	Mehdi O. et al [15] Instance based approach
Precision	86.4%	99.00%
Recall	91%	96%
F1-Score	88.6%	97%

According to the results, the performance of the study [15] is better than the methods that is proposed in this thesis. This is due to the usage of Google Similarity engine[16], which uses World Wide Web (WWW) as a database to provide automatic semantic of useful quality and Google Similarity calculates the semantic similarity score for the attributes. Moreover, Regular Expression tool [17] is used to ease the identification of text by describing it through pattern matching.

Finally, the comparisons that is done it is observed that the proposed methodology shows better performance results than the study [14]. However, the study [15] shows better performance results than the proposed methodology due to Google Similarity engine [16] and Regular Expressions tool [17].

Chapter 9

CONCLUSION

Schema matching is mostly used to locate and identify semantically related-target which eases the finding and matching of divergent and randomly scattered data sets. It is one valuable tool for data processing and schema integration.

This thesis proposes an automated database schema matching engine for matching database tables and form a table that contains the common data between these tables. This is done using the proposed methodology, it performs several steps; pre-processing the data, search for synonyms of the data entered, save the data to the database, match the data and classify the data. Moreover, one of the powerful characteristics of the proposed methodology is that, it can be automated hence, less time required to carry a particular task and more efficient if the task is more complex and if it is a larger scale task. The methodology used is trained like human and have the ability to predict the pattern of the data and it showed a very satisfactory results.

Furthermore, the proposed methodology can be further modified to increase the performance of the algorithms and methods used in this thesis, if CNN is used with google similarity tools or more powerful semantic tools and approaches helping the system to predict more synonyms.

REFERENCES

[1] Rahm, E., & Bernstein, P. (2001). A survey of approaches to automatic schema matching. *The VLDB Journal*, *10*(4), 334-350. doi: 10.1007/s007780100057.

[2] Black, P. E. (2019, March 18). Jaro-Winkler in Dictionary of Algorithms and Data Structures. Retrieved from https://www.nist.gov/dads/HTML/jaroWinkler.html

[3] Melnik, S., Garcia-Molina, H., & Rahm, E. Similarity flooding: a versatile graph matching algorithm and its application to schema matching. *Proceedings 18Th International Conference On Data Engineering*. doi: 10.1109/icde.2002.994702.

[4] Madhavan, J., Bernstein, P. A., & Rahm, E. (2001, September). Generic schema matching with cupid. In vldb (Vol. 1, pp. 49-58).

[5] Köpcke, H., & Rahm, E. (2010). Frameworks for entity matching: A comparison. *Data & Knowledge Engineering*, *69*(2), 197-210. doi: 10.1016/j.datak.2009.10.003.

[6] Berlin, J., & Motro, A. (2002). Database Schema Matching Using Machine Learning with Feature Selection. *Notes On Numerical Fluid Mechanics And Multidisciplinary Design*, 452-466. doi: 10.1007/3-540-47961-9_32.

[7] Li, W., & Clifton, C. (2000). SEMINT: A tool for identifying attribute correspondences in heterogeneous databases using neural networks. *Data & Knowledge Engineering*, *33*(1), 49-84. doi: 10.1016/s0169-023x(99)00044-0.

[8] Sorrentino, S., Bergamaschi, S., & Gawinecki, M. (2011). NORMS: An automatic tool to perform schema label normalization. *2011 IEEE 27Th International Conference On Data Engineering*. doi: 10.1109/icde.2011.5767952.

[9] Do, H., & Rahm, E. (2002). COMA — A system for flexible combination of schema matching approaches. *VLDB '02: Proceedings Of The 28Th International Conference On Very Large Databases*, 610-621. doi: 10.1016/b978-155860869-6/50060-3.

[10] n11.com - Alışverişin Uğurlu Adresi. (2020). Retrieved 6 February 2020, from https://www.n11.com/

[11] Malzemeler Veri Aktarımları - Tiger 3 Bilgi Deposu - Global Site. (2020). Retrieved 4 February 2020, from https://docs.logo.com.tr/pages/viewpage.action?pageId=22272929

[12] Mikro Tablolar. (2020). Retrieved 4 February 2020, from http://www.mye.com.tr/help/Library/Diger/DBYapisi_V15/tablo.htm

[13] ERK TEKNOLOJİ. (2020). Retrieved 25 February 2020, from http://www.erkteknoloji.com/

[14] Sahay, T., Mehta, A., & Jadon, S. (2019). Schema Matching using Machine Learning. arXiv preprint arXiv:1911.11543.

[15] Mehdi, O. A., Ibrahim, H., & Affendey, L. S. (2017). An approach for instance based schema matching with google similarity and regular expression. Int. Arab J. Inf. Technol., 14(5), 755-763.

[16] Cilibrasi, R. L., & Vitanyi, P. M. (2007). The google similarity distance. *IEEE Transactions on knowledge and data engineering*, *19*(3), 370-383.

[17] Friedl, J. E. (2006). *Mastering regular expressions*. " O'Reilly Media, Inc.".

APPENDICES

Appendix A: Training Data

Firm	Table name	Table field	Field description 1	Field description 2	Data type	Length	Mandatory (1/0)	Key
N11	Product	Product_seller_code			Varchar	25	1	Fk
Mikro	Stocks	Stock_code	Product code		Varchar	25	1	Pk
Logo	Items	Item_code	Material code		Varchar	25	1	Pk
Serotonin	Products	Product_code			Varchar	25	1	Fk
N11	Product	Title			Varchar	35	1	
Mikro	Stocks	Stock_name	Product name		Varchar	50		
Logo	Items	Items_name	Material description		Varchar	51	0	
Serotonin	Products	Product_name			Varchar	25	1	
N11	Product	Price	Product base price		Double	8	1	
Logo	Items	Price	Price		Double	8	0	
Serotonin	Product	Sales_price			Varchar	20	0	
Mikro	Stocks	Stock_retailtax	Retail tax rate		Tinyint			
Logo	Items	VAT	Tax		Double	8	0	
Serotonin	Products	Tax			Float	18	0	
N11	Product	Currency_type	Product list price currency		Varchar	30	1	
Mikro	Stocks	Stock_currency_type	Currency id		Tinyint			

Serotonin	Products	Currency_type			Varchar	5	0	
N11	Product	URL	Product official url		Varchar	30	1	
Logo	Items	Photo	Picture		Byte	1	0	
Serotonin	Products	Photo_name			Varchar	25	0	
N11	Product	Order	Product image display order		Varchar	30	1	
Serotonin	Products	Order			Int	11	0	
N11	Product	Product_condition	Product state		Varchar	30	1	
Logo	Items	Active	Registration status		Int	2	1	
Serotonin	Products	State			Varchar	2	0	
N11	Product	Discount	Product discount information		Varchar		1	
Serotonin	Products	Discount_price			Float	10.2	0	
N11	Product	Quantity	The amount of product		Varchar		1	
Mikro	Stocks	Sto_max_stock			Float			
Serotonin	Products	Stock_piece			Int	11	0	
N11	Product	Subtitle			Varchar		1	
N11	Product	Description	Product description information (can be html)		Varchar		1	
Serotonin	Products	Statement			Text		0	

N11	Product	Attribute	Name and value of the product properties are entered in the field		Varchar		0	
N11	Product	Id	Product category number		Varchar		1	
Serotonin	Products	Group_id			Varchar	20	0	
N11	Product	Sale_start_date	Product sales start date (dd / mm / yyyy)		Varchar		0	
N11	Product	Sale_end_date	Product sales end date (dd / mm / yyyy)		Varchar		0	
N11	Product	Date_of_production	Product production date (dd / mm / yyyy)		Varchar		0	
N11	Product	Expiration_date	Product expiration date (dd / mm / yyyy)		Varchar		0	
Logo	Items	Shelf_date	Expiration date		Int	2	0	
Serotonin	Products	Expiration_date			Varchar	50	0	
N11	Product	Preparing_day	Production time to ship (in days)		Varchar		1	
Mikro	Stocks	Sto_order	Order time (days)		Int			
N11	Product	Shipment_template	Delivery template name				1	

N11	Product	Seller_store_code	Store product code		Varchar		0	
Mikro	Stocks	store_product_code	Store product code		Varchar	25		
N11	Product	Option_price	List price of the product unit		Varchar		0	
N11	Product	Bundle	Bundle products		Varchar		0	
N11	Product	MPN	Production manufacturer part number		Varchar		0	
N11	Product	GTIN	Production global trading item number		Varchar		0	
Logo	Items	Card_type	Material registration type		Int	2	1	
Logo	Items	Grp_code	Material group code		Varchar	17	0	
N11	Product	GCIN	Production global commercial item number		Varchar		0	
Mikro	Stocks	Sto_producer_code	Manufacturer code		Varchar	25		
Logo	Items	Producer_code	Manufacturer code		Varchar	25	0	
Logo	Items	Spec_code	Special code		Varchar	11	0	
Logo	Items	Authority_code	Authorization code		Varchar	11	0	
Logo	Items	Class_type	Material class type		Int	2	0	

Logo	Items	PURCHBRWS	Place of use - purchasing		Int	2	0	
Logo	Items	Sales_BRWS	Place of use - sales and distribution		Int	2	0	
Logo	Items	MTRLBRWS	Place of use - store management		Int	2	0	
Logo	Items	Track_type	Track type		Byte	1	0	
Logo	Items	Inv_tracking	Product inventory tracking		Byte	1	0	
Logo	Items	Tool	Tool		Byte	1	0	
Logo	Items	AUTOINCSL	Automatically increase product serial number		Byte	1	0	
Logo	Items	Div_lot_size	Lot size can be split		Byte	1	0	
Mikro	Stocks	Sto_shelf_life	Shelf life		Int			
Logo	Items	Shelf_life	Shelf life		Double	8	0	
Logo	Items	Depr_type	Depreciationtype		Int	2	0	
Logo	Items	Depr_rate	Depreciationrate		Double	8	0	
Logo	Items	Depr_dur	Depreciationduration		Int	2	0	
Logo	Items	Salvage_val	Salvagevalue		Double	8	0	
Logo	Items	Reval_flag	Revaluation		Byte	1	0	
Logo	Items	Rev_depr_flag	Valuationdepreciation		Byte	1	0	
Logo	Items	Part_dep	Part depreciation		Int	1	0	
Logo	Items	Depr_type2	Depreciation type2		Double	2	0	
Logo	Items	Depr_rate2	Depreciation rate2		Int	8	0	

Logo	Items	Depr_dur2	Depreciation duration2		Real	2	0	
Logo	Items	Reval_flag2	Revaluation 2		Byte	1	0	
Logo	Items	Rev_depr_flag2	Alternative valuation depreciation		Byte	1	0	
Logo	Items	Part_dep2	Part depreciation2		Byte	1	0	
Logo	Items	approved	Approved		Byte	1	0	
Logo	Items	Dist_amount	Distributed amount		Double	8	0	
Logo	Items	Created_by	Created by		Int	2	0	
Logo	Items	Created_date	Created date		Longint	4	0	
Logo	Items	Created_hour	Created hour		Int	2	0	
Logo	Items	Created_min	Created minute		Int	2	0	
Logo	Items	Created_sec	Created secound		Int	2	0	
Logo	Items	Modified_by	Modifed		Int	2	0	
Logo	Items	Modified_date	Modifed date		Longint	4	0	
Serotonin	Products	Date_of_update			Varchar	30	0	
Logo	Items	Modified_hour	Modifedhour		Int	2	0	
Logo	Items	Modified_min	Modifedminute		Int	2	0	
Logo	Items	Modified_sec	Modifed seconds		Int	2	0	
Logo	Items	Site_id	Data center		Int	2	0	
Logo	Items	Data_ref	Data reference		Longint	4	0	
Logo	Items	Univid	Out of use		Varchar	25	0	

Logo	Items	Dist_lot_units	Lot size can be distributed		Byte	1	0	
Logo	Items	Comb_lot_units	Lot sizes combineable		Byte	1	0	
Logo	Items	Lot_sizing_mtd	Lot determination method		Int	2	0	
Logo	Items	Fixed_lot_size	Fixed lot size		Double	8	0	
Logo	Items	Yield	Yield		Double	8	0	
Logo	Items	Min_order_qty	Minimum order quantity		Double	8	0	
Logo	Items	Max_order_qty	Max order quantity		Double	8	0	
Logo	Items	Mult_order_qty	Multi order quantity		Double	8	0	
Logo	Items	Dominant_code	Materialcard code		Varchar	25	0	
Logo	Items	Class_type	Class type		Int	2	0	
Logo	Items	Class_code	Class code		Varchar	25	0	
Logo	Items	Dist_point	Distribution point		Double	8	0	
Logo	Items	Can_use_intrns	Used in movements (1: true 2: false)		Byte	1	0	
Mikro	Stocks	Sto_rec_no			Integer identity			
Mikro	Stocks	Sto_rec_id_dbcno			Smallint			
Mikro	Stocks	Sto_rec_id_recno			Integer			

Mikro	Stocks	Sto_spec_rec_no			Integer			
Mikro	Stocks	Sto_cancel			Bit			
Mikro	Stocks	Sto_file_id			Smallint			
Mikro	Stocks	Sto_hidden			Bit			
Mikro	Stocks	Sto_locked			Bit			
Mikro	Stocks	Sto_changed			Bit			
Mikro	Stocks	Sto_check_sum			Integer			
Mikro	Stocks	Sto_create_user			Smallint			
Mikro	Stocks	Sto_create_date			Datetime			
Mikro	Stocks	Sto_last_up_user			Smallint			
Mikro	Stocks	Sto_last_up_date			Datetime			
Mikro	Stocks	Sto_special1			Varchar	4		
Mikro	Stocks	Sto_special2			Varchar	4		
Mikro	Stocks	Sto_special3			Varchar	4		
Mikro	Stocks	Sto_short_name	Sto short name		Varchar	25		
Mikro	Stocks	Sto_foreign_name	Sto foreign name		Varchar	50		
Mikro	Stocks	Sto_seller_current_code	Seller current code	See. Table current accounts	Varchar	25		

Mikro	Stocks	Sto_type	Product type	0: commercial goods 1: first article 2: intermediate product 3: semi-finished product 4: product 5: side product 6: operating material 7: consumption material 8: spare part 9: fuel stock 10: installation prescription product 11: basic raw material	Tinyint			
Mikro	Stocks	Sto_following_details	Following details	0: no detail tracking 1: party basis 2: party lot basis 3: serial number basis 4: 5 on the basis of bond	Tinyint			
Mikro	Stocks	Sto_unit1_name	Unit name		Varchar	10		
Mikro	Stocks	Sto_unit1_coefficient	Unit1 coefficient		Float			
Mikro	Stocks	Sto_unit1_weight	Unit net weight (kg)		Float			
Mikro	Stocks	Sto_unit1_width	Unit width (mm)		Float			
Mikro	Stocks	Sto_unit1_length	Unit length (mm)		Float			
Mikro	Stocks	Sto_unit1_height	Unit height (mm)		Float			
Mikro	Stocks	Sto_unit1_tare	Unit1 tare		Float			

Mikro	Stocks	Sto_unit2name	Unit name		Varchar	10		
Mikro	Stocks	Sto_unit2_coefficient	Unit coefficient		Float			
Mikro	Stocks	Sto_unit2_weight	Unit net weight (kg)		Float			
Mikro	Stocks	Sto_unit2_width	Unit width (mm)		Float			
Mikro	Stocks	Sto_unit2_length	Unit length (mm)		Varchar	10		
Mikro	Stocks	Sto_unit2_height	Unit height (mm)		Float			
Mikro	Stocks	Sto_unit2_tare			Float			
Mikro	Stocks	Sto_unit3_name	Unit name		Float			
Mikro	Stocks	Sto_unit3_coefficient	Unit coefficient		Float			
Mikro	Stocks	Sto_unit3_weight	Unit net weight (kg)		Float			
Mikro	Stocks	Sto_unit3_width	Unit width (mm)		Float			
Mikro	Stocks	Sto_unit3_length	Unit length (mm)		Varchar	10		
Mikro	Stocks	Sto_unit3_height	Unit height (mm)		Float			
Mikro	Stocks	Sto_unit3_tare			Float			

Mikro	Stocks	Sto_unit4_name	Unit name		Float			
Mikro	Stocks	Sto_unit4_coefficient	Unit coefficient		Float			
Mikro	Stocks	Sto_unit4_weight	Unit net weight (kg)		Float			
Mikro	Stocks	Sto_unit4_width	Unit width (mm)		Float			
Mikro	Stocks	Sto_unit4_length	Unit length (mm)		Varchar	40		
Mikro	Stocks	Sto_unit4_height	Unit height (mm)		Varchar	40		
Mikro	Stoklar	Sto_unit4_tara			Nvarchar(40)			
Mikro	Stoklar	Sto_acc_code	Product acc account code		Varchar	40		
Logo	Items	Acc_code	Acc account code		Varchar	25	0	
Serotonin	Product	Acc_code			Varchar	25	0	
Mikro	Stoklar	Sto_acc_return_code	Sto_acc_return_code		Varchar	40		
Mikro	Stoklar	Sto_acc_purchase_acc_code	Stok acc. Purchasing code		Varchar	40		
Mikro	Stoklar	Sto_acc_satiadacccode	Stok acc. Purchasing return code		Varchar	40		
Mikro	Stocks	Sto_acc_discount_code	Product acc. Discount code		Varchar	40		

Mikro	Stocks	Sto_acc_purchasing_discount_code	Product acc. Purchasing discount code		Varchar	40		
Mikro	Stocks	Sto_acc_sales_costs_code	Product acc. Sales costs code		Varchar	40		
Mikro	Stocks	Sto_acc_overseas_sales_code	Product acc. Overseas sales code		Varchar	40		
Mikro	Stocks	Sto_acc_extra_charges_code	Product acc. Extra charges code		Varchar	40		
Mikro	Stocks	Sto_invastment_promo_acc_code	Investment promotion of acc. Code		Varchar	40		
Mikro	Stocks	Sto_stores_sales_acc_code	Stores sales acc. Code		Varchar	40		
Mikro	Stocks	Sto_sales_cost_stores_acc_code	Cost of sales between stores of acc. Code		Varchar	40		
Mikro	Stocks	Sto_bagortsatacccode	Partners, depending on sales acc. Code		Varchar	40		
Mikro	Stocks	Sto_bagortsatiadacccode	Return sales to affiliates acc. Code		Varchar	40		
Mikro	Stocks	Sto_bagortsatiskacccode	Depending on the partner sales discount acc. Code		Varchar	40		
Mikro	Stocks	Sto_diff_sales_price_acc_code	The difference in the sale price acc. Code		Varchar	40		

Mikro	Stocks	Sto_cost_export_sales_acc_code	The cost of export sales acc. Code		Float			
Mikro	Stocks	Sto_affiliate_sales_cost_acc_code	Affiliate sales cost acc. Code		Float			
Mikro	Stocks	Sto_zero_paid_cost_sales_acc_code	Zero paid cost of sales acc. Code		Float			
Mikro	Stocks	Sto_profit	Profit rate		Float			
Mikro	Stocks	Sto_min_stock	The minimum product level		Tinyint			
Mikro	Stocks	Sto_order_stock	Product order level		Tinyint			
Mikro	Stocks	Sto_max_stock	Maximum product level		Smallint			
Mikro	Stocks	Sto_given_order_unit	Given order unit		Tinyint			
Mikro	Stocks	Sto_received_order_unit	Received order unit		Tinyint			
Mikro	Stocks	Sto_order_time	Order time (days)		Varchar	25		
Mikro	Stocks	Sto_retail_rate	Retail vat rate		Tinyint			
Mikro	Stocks	Sto_wholesale_rate	Wholesale vat rate		Tinyint			
Mikro	Stocks	Sto_warehouse_code	Warehouse address		Tinyint			

Mikro	Stocks	Sto_electronic_label_type	Electronic label type	0: standard label 1: small sticker 2: fruit and vegetable label	Tinyint			
Mikro	Stocks	Sto_shelf_label	Shelf label	0:no 1:yes	Tinyint			
Mikro	Stocks	Sto_label_account	Print a label?	0:did not print 1:print	Tinyint			
Mikro	Stocks	Sto_sales_stops	Sales stop?	0: did not stop 1: stop	Bit			
Mikro	Stocks	Sto_stop_order	Stop order?	0: did not stop 1: stop	Bit			
Mikro	Stocks	Sto_stop_accepted_goods	Will you accept the goods?	0: did not stop 1: stop	Bit			
Mikro	Stocks	Sto_accepted_day1	Goods acceptance day	Monday	Bit			
Mikro	Stocks	Sto_accepted_day2	Goods acceptance day	Tuesday	Bit			
Mikro	Stocks	Sto_accepted_day3	Goods acceptance day	Wednesday	Bit			
Mikro	Stocks	Sto_accepted_day4	Goods acceptance day	Thursday	Bit			
Mikro	Stocks	Sto_accepted_day5	Goods acceptance day	Friday	Bit			
Mikro	Stocks	Sto_accepted_day6	Goods acceptance day	Saturday	Bit			
Mikro	Stocks	Sto_accepted_day7	Goods acceptance day	Sunday	Bit			
Mikro	Stocks	Sto_orders_day1	Order days	Monday	Bit			

Mikro	Stocks	Sto_orders_day2	Order days	Tuesday	Bit			
Mikro	Stocks	Sto_orders_day3	Order days	Wednesday	Bit			
Mikro	Stocks	Sto_orders_day4	Order days	Thursday	Bit			
Mikro	Stocks	Sto_orders_day5	Order days	Friday	Bit			
Mikro	Stocks	Sto_orders_day6	Order days	Saturday	Bit			
Mikro	Stocks	Sto_orders_day7	Order days	Sunday	Smallint			
Mikro	Stocks	Sto_discount_can't_done	Discounts can't be done?	0: yes 1: no	Varchar	25		
Mikro	Stocks	Sto_inliquidation	Short-lived provisional all?	0: yes 1: no	Varchar	25		
Mikro	Stocks	Sto_sub_group_no	Sub group number		Varchar	25		
Mikro	Stocks	Sto_category_code	Product category code		Varchar	25		
Mikro	Stocks	Sto_product_officier_code	Product officer code	See. Table staff	Varchar	25		
Mikro	Stocks	Sto_invent_subgroup_code	Inventory subgroup code	See. Table stock sub groups	Varchar	25		
Mikro	Stocks	Sto_parent_group_code	Product of the parent group code	See. Table stock main groups	Varchar	25		
Mikro	Stocks	Sto_producer_code	Manufacturer code	See. Table stock_producers	Varchar	25		

Mikro	Stocks	Sto_sector_code	Sector code	See. Table stock_sectors	Varchar	25		
Mikro	Stocks	Sto_accgroup_code	Acc. Group code	See. Table stock_acc_groups	Varchar	25		
Mikro	Stocks	Sto_packaging_code	Packaging code	See. Table stock_packagins	Varchar	25		
Mikro	Stocks	Sto_brand_code	Brand code	See. Table stock_brands	Varchar	25		
Mikro	Stocks	Sto_size_code	Product size code	See. Table stock_size_definitions	Varchar	25		
Mikro	Stocks	Sto_color_code	Color code	See. Table stock_color_definitions	Varchar	25		
Mikro	Stocks	Sto_model_code	The model code	See. Table stock_model_definitions	Varchar	25		
Mikro	Stocks	Sto_season_code	Season code	See. Table stock_year_season_definitions	Varchar	25		
Mikro	Stocks	Sto_raw_material_code	Raw material code	See. Table stock_main_raw_material	Varchar	25		
Mikro	Stocks	Sto_premium_code	Premium code		Varchar	25		
Mikro	Stocks	Sto_quality_control_code	Quality control code	See. Table stock_quality_control_definitions	Varchar	10		

Mikro	Stocks	Sto_package_code	Package code	See. Table stock_package_definitions	Bit			
Mikro	Stocks	Sto_positionflag_code	Position the flag code		Bit			
Mikro	Stocks	Sto_acccode_discard	Product acc code discarded		Bit			
Mikro	Stocks	Sto_safe_wighed	Goods weighed in the safe?	0: yes 1: no	Bit			
Mikro	Stocks	Sto_size_followup	Size detailed?	0: yes 1: no	Bit			
Mikro	Stocks	Sto_colordetail	Color detailed?	0: yes 1: no	Bit			
Mikro	Stocks	Sto_quantity_decimal	Does it produce decimal?		Varchar	25		
Mikro	Stocks	Sto_passive	Active/passive	0: passive 1: active	Float			
Mikro	Stocks	Sto_decreasing_stock	Product may fall to negative?		Varchar	25		
Mikro	Stocks	Sto_custom_tax_statistical_position	Customs identification statistics position number		Float			
Mikro	Stocks	Sto_point			Tinyint			
Mikro	Stocks	Sto_commission_service_code	The commission service code		Float			
Mikro	Stocks	Sto_commission_rate	Commission rate		Tinyint			

Mikro	Stocks	Sto_otv_application	Ötv application	0: ötv no 1: receipt from the amount 2: meet the percentage 3: from the amount on sale 4: sales percentage 5: receipt and sales amount 6: receipt and sales percentage	Tinyint			
Mikro	Stocks	Sto_otv_amount	Ötv amount		Float			
Mikro	Stocks	Sto_otv_list	Ötv type	0:no 1: ötv1 2: ötv2 3: ötv3 4: ötv4 5: ötv3a 6: ötv3b 7: ötv3c	Smallint			
Mikro	Stocks	Sto_otv_unit	Product ötv unit		Tinyint			
Mikro	Stocks	Sto_premium_rate	Premium-rate		Float			
Mikro	Stocks	Sto_warranty_period	The predicted warranty period		Float			
Mikro	Stocks	Sto_warranty_period_type	Type of warranty period	0: month 1: day 2: year	Float			
Mikro	Stocks	Sto_fiber_no	Fiber number		Tinyint			
Mikro	Stocks	Sto_standard_cost	Standard cost		Bit			
Mikro	Stocks	Sto_picking_cash_amount	Product picking cash amount		Float			
Mikro	Stocks	Sto_communication_tax_application	Is there special communication tax application?	0: no 1: yes	Bit			
Mikro	Stocks	Sto_z_report_stocks	Z report?		Varchar	25		

Mikro	Stocks	Sto_max_discount_rate	The maximum discount rate		Tinyint			
Mikro	Stocks	Sto_detail_tracking_inwarehouse control	Detail the tracking of warehouse control?		Varchar	25		
Mikro	Stocks	Sto_complementary_code	Complementary code		Float			
Mikro	Stocks	Sto_auto_barcode_login_form	Automatic barcode login form	0: automatic barcode creation 1: automatic barcode to be created according to the detail tracking 2: create barcode for each entry record	Float			
Mikro	Stocks	Sto_auto_barcode_code_strucutre	Automatic barcode code structure		Float			
Mikro	Stocks	Sto_case_discount_rate	Case discount rate		Float			
Mikro	Stocks	Sto_cash_discount_amount	Cash discount amount		Tinyint			
Mikro	Stocks	Sto_revenue_share	Revenue share		Varchar	25		
Mikro	Stocks	Sto_summary_communication_tax_ammount	Summary communication tax amount		Tinyint			
Mikro	Stocks	Sto_summary_communication_tax_type	Summary communication tax type	0: none 1: sct 2:5035 nambered by the low of sct	Tinyint			

Mikro	Stocks	Sto_expense_code	Expense code		Bit			
Mikro	Stocks	Sto_sct	(summary communication tax) sct		Smallint			
Mikro	Stocks	Sto_deductions_type	Deductions type	0: withholding 1: withholding 31 2: withholding 91 3: withholding 21 4: withholding 32 5: withholding 61 6: withholding 45 7: withholding full	Bit			
Mikro	Stocks	Sto_expirdat_fl	Is there an expiry date?		Varchar	40		
Mikro	Stocks	Sto_balance_expirdat	Balance expiration date		Varchar	40		
Mikro	Stocks	Sto_installable_checkout	Installable at checkout?		Varchar	40		
Mikro	Stocks	Sto_ifrsdifference_code	Difference of (international financial reporting standards) ifrs acc. Code		Varchar	40		
Mikro	Stocks	Sto_refund_ifrsdifference_code	Refund ifrs difference acc. Code		Varchar	40		

Mikro	Stocks	Sto_domestic_sales_ifrsdifference_code	Domestic sales ifrs difference acc. Code		Varchar	40		
Mikro	Stocks	Sto_sales_refund_ifrsdifference_code	Purchasing refund ifrs difference acc. Code		Varchar	40		
Mikro	Stocks	Sto_sales_discount_ifrsdifference_code	Sales discount ifrs difference acc. Code		Varchar	40		
Mikro	Stocks	Sto_purchase_discount_ifrsdifference_code	Purchase discount ifrs difference acc. Code		Varchar	40		
Mikro	Stocks	Sto_sales_cost_ifrsdifference_code	Sales cost ifrs difference acc. Code		Varchar	40		
Mikro	Stocks	Sto_overseas_sales_ifrsdifference_code	Overseas sales ifrs difference acc. Code		Varchar	40		
Mikro	Stocks	Sto_additional_expenses_ifrsdifference_code	Additional expenses ifrs difference acc. Code		Varchar	40		
Mikro	Stocks	Sto_investment_incentive_ifrsdifference_code	Investment incentive ifrs difference acc. Code		Varchar	40		

Mikro	Stocks	Sto_warehouses_inter_sales_ifrsdifference_code	Warehouses inter-sales ifrs difference acc. Code		Varchar	40		
Mikro	Stocks	Sto_sales_cost_warehouses_ifrsdifference_code	Cost of sale between warehouses ifrs difference acc. Code		Varchar	40		
Mikro	Stocks	Sto_partnership_sales_ifrsdifference_code	Sales of partnership ifrs difference acc. Code		Varchar	40		
Mikro	Stocks	Sto_sales_return_ifrsdifference_code	Sales return to affiliated companies ifrs difference acc. Code		Varchar	40		
Mikro	Stocks	Sto_discounted_sales_ifrsdifference_code	Discounted sale to affiliated partnerships ifrs difference acc. Code		Varchar	40		
Mikro	Stocks	Sto_diff_sales_price_ifrsdifference_code	Sales price differential ifrs difference acc. Code		Varchar	40		
Mikro	Stocks	Sto_overseas_sales_cost_ifr	Overseas sales cost ifrs difference acc. Code		Varchar	40		

		sdifference_code						
Mikro	Stocks	Sto_partnership_cost_sales_ifrsdifference_code	Partnerships based on cost of sales ifrs difference acc. Code		Varchar	40		
Mikro	Stocks	Sto_zero_cost_sales_ifrsdifference_kod	Zero paid cost of sales ifrs difference acc. Code		Float			
Mikro	Stocks	Sto_costt_prod_ifrsdifference_code	Cost of production of ifrs difference acc. Code		Bit			
Mikro	Stocks	Sto_prod_capacity_ifrsdifference_code	Production capacity of ifrs difference acc. Code		Smallint			
Mikro	Stocks	Sto_impairment_ifrsdifference_code	Impairment of ifrs difference acc. Code		Smallint			
Mikro	Stocks	Sto_percentage_of_content	Percentage of content		Smallint			
Mikro	Stocks	Sto_will_sent_to_web	Will it be sent to the web		Bit			
Mikro	Stocks	Sto_min_stock_daily_info	Daily information for minimum leveling operation		Tinyint			
Mikro	Stocks	Sto_order_stock_daily_info	Daily information for order leveling operation		Tinyint			

Mikro	Stocks	Sto_max_stock_daily_info	Daily information for maximum leveling operation		Smallint			
Mikro	Stocks	Sto_levelling_operation_evaluation	Will the assessment of the leveling operation be carried out?		Bit			
Mikro	Stocks	Sto_otv_deduction_type	Otv deduction type	0:no deduction 1: withholding	Tinyint			
Mikro	Stoklar	Sto_reso_plan_evaluation	Will be evaluated in the resource planning operation?	0: true1: false	Tinyint			

Appendix B: System Final Schema

Field name	Description_1	Description_2	Syn
Acc_code	Product acc account code		Acc_code, acc_id
Acc_code_discard	Product acc code discarded		Acc_code_discard
Acc_discount_code	Product acc. Discount code		Acc_discount_code
Acc_extra_charges_code	Product acc. Extra charges code		Acc_extra_charges_code
Acc_group_code	Acc. Group code	See. Table stock_acc_groups	Acc_group_code
Acc_overseas_sales_code	Product acc. Overseas sales code		Acc_overseas_sales_code
Acc_purchase_acc_code	Stok acc. Purchasing code		Acc_purchase_acc_code
Acc_purchasing_discount_code	Product acc. Purchasing discount code		Acc_purchasing_discount_code
Acc_return_code	Acc return code		Acc_return_code

Acc_sales_costs_code	Product acc. Sales costs code		Acc_sales_costs_code
Accepted_day1	Goods acceptance day	Monday	Accepted_day1
Accepted_day2	Goods acceptance day	Tuesday	Accepted_day2
Accepted_day3	Good acceptance day	Wednesday	Accepted_day3
Accepted_day4	Good acceptance day	Thursday	Accepted_day4
Accepted_day5	Good acceptance day	Friday	Accepted_day5
Accepted_day6	Good acceptance day	Saturday	Accepted_day6
Accepted_day7	Good acceptance day	Sunday	Accepted_day7
Active	Registration status		Active
Affiliate_sales_cost_acc_code	Affiliate sales cost acc. Code		Affiliate_sales_cost_acc_code
Approved	Approved		Approved, accepted
Attribute	Name and value of the product properties are entered in the field		Attribute
Authority_code	Authorization code		Authority_code
Auto_bar_code_code_structure	Automatic barcode code structure		Auto_bar_code_code_structure

Auto_bar_code_login_form	Automatic barcode login form	0: automatic barcode creation 1: automatic barcode to be created according to the detail tracking 2: create barcode for each entry record	Auto_bar_code_login_form
Automatically_increase_product_serial_number	Automatically increase product serial number		Automatically_increase_product_serial_number
Balance_expiry_date	Balance expiration date		Balance_expiry_date
Brand_code	Brand code	See. Table stock_brands	Brand_code
Bundle	Bundle products		Bundle, package
Can_use_interns	Used in movements (1: true 2: false)		Can_use_interns
Cancel			Cancel, drop
Case_discount_rate	Case discount rate		Case_discount_rate
Cash_discount_amount	Cash discount amount		Cash_discount_amount
Category_code	Product category code		Category_code
Changed			Changed, improved, altered
Check_sum			Check_sum
Class_code	Class code		Class_code

Class_type	Material class type		Class_type
Colour_code	Color code	See. Table stock_color_definitions	Colour_code
Colour_detail	Color detailed?	0: yes 1: no	Colour_detail
Comb_lot_units	Lot sizes combineable		Comb_lot_units
Commission_rate	Commission rate		Commission_rate
Commission_service_code	The commission service code		Commission_service_code
Communication_tax_application	Is there special communication tax application?	0: no 1: yes	Communication_tax_application
Complementary_code	Complementary code		Complementary_code
Condition	Product state		Condition, state
Cost_export_sales_acc_code	The cost of export sales acc. Code		Cost_export_sales_acc_code
Create_user			Create_user
Created_by	Created by		Created_by
Created_date	Created date		Created_date

Created_hour	Created hour		Created_hour
Created_min	Created minute		Created_min
Created_sec	Created second		Created_sec
Currency_type	Product list price currency		Currency_type
Custom_tax_statistical_position	Customs identification statistics position number		Custom_tax_statistical_position
Data_ref	Data reference		Data_ref
Date_of_production	Product production date (dd / mm / yyyy)		Date_of_production
Date_of_update			Date_of_update
Decreasing_stock	Product may fall to negative?		Decreasing_stock
Deductions_type	Deductions type	0: withholding 1: withholding 31 2: withholding 91 3: withholding 21 4: withholding 32 5: withholding 61 6: withholding 45 7: withholding full	Deductions_type
Depr_dur2	Depreciation duration2		Depr_dur2

Depr_rate	Depreciationrate		Depr_rate
Depr_rate2	Depreciation rate2		Depr_rate2
Depr_type	Depreciationtype		Depr_type
Depr_type2	Depreciation type2		Depr_type2
Depreciation_duration	Depreciation duration		Depreciation_duration
Description	Product description information (can be html)		Description
Detail_tracking_in_warehouse_control	Detail the tracking of warehouse control?		Detail_tracking_in_warehouse_control
Diff_sales_price_acc_code	The difference in the sale price acc. Code		Diff_sales_price_acc_code
Discount	Product discount information		Discount
Discount_can't_done	Discounts can't be done?	0: yes 1: no	Discount_can't_done
Discount_price			Discount_price
Dist_amount	Distributed amount		Dist_amount
Dist_lot_units	Lot size can be distributed		Dist_lot_units
Divided_lot_size	Lot size can be split		Divided_lot_size

Dominant_code	Materialcard code		Dominant_code, dominant_id
Electronic_label_type	Electronic label type	0: standard label 1: small sticker 2: fruit and vegetable label	Electronic_label_type
Expense_code	Expense code		Expense_code
Expiration_date	Product expiration date (dd / mm / yyyy)		Expiration_date
Expiry_date	Is there an expiry date?		Expiry_date, end_date
Fibre_no	Fiber number		Fibre_no, fibre_code
File_id			File_id, file_code
Fixed_lot_size	Fixed lot size		Fixed_lot_size
Following_details	Following details	0: no detail tracking 1: party basis 2: party lot basis 3: serial number basis 4: 5 on the basis of bond	Following_details
Foreign_name			Foreign_name, foreign_label, foreign_tag
Given_order_unit	Given order unit		Given_order_unit
Group_id			Group_id
Hidden			Hidden, concealed
Id	Product category number		Id, code

In_liquidation	Short-lived provisional all?	0: yes 1: no	In_liquidation
Installable_at_checkout	Installable at checkout		Installable_at_checkout
Inventory_subgroup_code	Inventory subgroup code	See. Table stock sub groups	Inventory_subgroup_code
Inventory_tracking	Product inventory tracking		Inventory_tracking
Investment_promo_acc_code	Investment promotion of acc. Code		Investment_promo_acc_code
Label_account	Print a label?	0:did not print 1:print	Label_account
Last_up_date			Last_up_date
Last_up_user			Last_up_user
Levelling_operation_evaluation	Will the assessment of the leveling operation be carried out?		Levelling_operation_evaluation
Locked			Locked
Lot_sizing_mtd	Lot determination method		Lot_sizing_mtd
Max_discount_rate	The maximum discount rate		Max_discount_rate
Max_order_qty	Max order quantity		Max_order_qty
Max_stock			Max_stock

Max_stock_daily_info	Daily information for maximum leveling operation		Max_stock_daily_info
Max_stock_level	Maximum product level		Max_stock_level
Min_order_qty	Minimum order quantity		Min_order_qty
Min_stock	The minimum product level		Min_stock
Min_stock_daily_info	Daily information for minimum leveling operation		Min_stock_daily_info
Model_code	The model code	See. Table stock_model_definitions	Model_code
Modified_by	Modifed		Modified_by
Modified_date	Modifed date		Modified_date
Modified_hour	Modified hour		Modified_hour
Modified_min	Modifedminute		Modified_min
Modified_sec	Modifed seconds		Modified_sec
Mult_order_qty	Multi order quantity		Mult_order_qty
Name			Name, label, tag

O_t_v_amount	Ötv amount		O_t_v_amount
O_t_v_application	Ötv application	0:ötv no 1:receipt from the amount 2:meet the percentage 3:from the amount on sale 4:sales percentage 5:receipt and sales amount 6:receipt and sales percentage	O_t_v_application
O_t_v_deduction_type	Otv deduction type	0:no deduction 1:withholding	O_t_v_deduction_type
O_t_v_list	Ötv type	0:no 1:ötv1 2:ötv2 3:ötv3 4:ötv4 5:ötv3a 6:ötv3b 7:ötv3c	O_t_v_list
O_t_v_unit	Product ötv unit		O_t_v_unit
Option_price	List price of the product product unit		Option_price
Order	Product image display order		Order
Order_stock_daily_info	Daily information for order leveling operation		Order_stock_daily_info
Order_time	Order time (days)		Order_time
Orders_day1	Order days	Monday	Orders_day1

Orders_day2	Order days	Tuesday	Orders_day2
Orders_day3	Order days	Wednesday	Orders_day3
Orders_day4	Order days	Thursday	Orders_day4
Orders_day5	Order days	Friday	Orders_day5
Orders_day6	Order days	Saturday	Orders_day6
Orders_day7	Order days	Sunday	Orders_day7
Package_code	Package code	See. Table stock_package_definitions	Package_code
Packaging_code	Packaging code	See. Table stock_packagins	Packaging_code
Parent_group_code	Product of the parent group code	See. Table stock main groups	Parent_group_code
Parent_group_code	Product of the parent group code	See. Table stock main groups	Parent_group_code
Part_dep	Part depreciation		Part_dep
Partdep2	Part depreciation2		Partdep2
Passive	Active/passive	0: passive 1: active	Passive
Percentage_of_content	Percentage of content		Percentage_of_content
Photo			Photo
Photo_name			Photo_name

Picking_cash_amount	Product picking cash amount		Picking_cash_amount
Point			Point
Position_flag_code	Position the flag code		Position_flag_code
Premium_code	Premium code		Premium_code
Premium_rate	Premium-rate		Premium_rate
Preparing_day	Production time to ship (in days)		Preparing_day
Price	Product base price		Price
Producer_code	Manufacturer code		Producer_code
Product_code			Product_code, stock_code, item_code
Product_location_store_management	Place of use - store management		Product_location_store_management
Product_officer_code	Product officer code	See. Table staff	Product_officer_code
Product_place_of_purchase	Place of use - purchasing		Product_place_of_purchase
Product_place_of_sales_and_distribution	Place of use - sales and distribution		Product_place_of_sales_and_distribution
Product_type	0: commercial goods 1: first article 2:		Product_type

	intermediate product 3: semi-finished product 4: product 5: side product 6: operating material 7: consumption material 8: spare part 9: fuel stock 10: installation prescription product 11: basic raw material		
Production_global_commercial_item_number	Production global commercial item number		Production_global_commercial_item_number
Production_global_trading_item_number	Production global trading item number		Production_global_trading_item_number
Production_manufacturer_part_number	Production manufacturer part number		Production_manufacturer_part_number
Profit	Profit rate		Profit
Quality_control_code	Quality control code	See. Table stock_quality_control_definitions	Quality_control_code
Quantity	The amount of product		Quantity, amount

Quantity_decimal	Does it produce decimal?		Quantity_decimal
Raw_material_code	Raw material code	See. Table stock_main_raw_material	Raw_material_code
Rec_id_rec_no			Rec_id_rec_no
Rec_no			Rec_no
Received_order_unit	Received order unit		Received_order_unit
Resource_plan_evaluation	Will be evaluated in the resource planning operation?	0:true1:false	Resource_plan_evaluation
Retail_rate	Retail vat rate		Retail_rate
Rev_depr_flag	Revaluation depreciation		Rev_depr_flag
Rev_depr_flag2	Alternative valuation depreciation		Rev_depr_flag2
Revaluation_flag	Revaluation		Revaluation_flag
Revaluation_flag2	Revaluation 2		Revaluation_flag2
Revenue_share	Revenue share		Revenue_share
Safe_weighed	Goods weighed in the safe?	0: yes 1: no	Safe_weighed

Sales_cost_stores_acc_code	Cost of sales between stores of acc. Code		Sales_cost_stores_acc_code
Sales_end_date	Product sales end date (dd / mm / yyyy)		Sales_end_date
Sales_price			Sales_price
Sales_start_date	Product sales start date (dd / mm / yyyy)		Sales_start_date
Sales_stop	Sales stop?	0:did not stop 1:stop	Sales_stop
Salvage_value	Salvage value		Salvage_value
Season_code	Season code	See. Table stock_year_season_definitions	Season_code
Sector_code	Sector code	See. Table stock_sectors	Sector_code
Seller_code			Seller_code
Seller_store_code	Store product code		Seller_store_code
Shelf_label	Shelf label	0:no 1:yes	Shelf_label
Shelf_life	Shelf life		Shelf_life
Shipment_template	Delivery template name		Shipment_template
Short_name	Short name		Short_name

Site_id	Data center		Site_id
Size_code	Product size code	See. Table stock_size_definitions	Size_code
Size_followup	Size detailed?	0: yes 1: no	Size_followup
Spec_code	Special code		Spec_code
Spec_rec_no			Spec_rec_no
Special1			Special1
Special2			Special2
Standard_cost	Standard cost		Standard_cost
State			State
Statement			Statement
Stock_name	Product name		Stock_name, product_name, item_name
Stock_order	Order time (days)		Stock_order
Stock_pieces			Stock_pieces
Stock_retail_tax	Retail tax rate		Stock_retail_tax
Stop_accepted_goods	Will you accept the goods?	0:did not stop 1:stop	Stop_accepted_goods
Stop_order	Stop order?	0:did not stop 1:stop	Stop_order
Store_product_code	Store product code		Store_product_code

Stores_sales_acc_code	Stores sales acc. Code		Stores_sales_acc_code
Sub_group_no	Sub group number		Sub_group_no
Subtitle			Subtitle
Summary_communication_tax	(summary communication tax) sct		Summary_communication_tax
Summary_communication_tax_amount	Summary communication tax amount		Summary_communication_tax_amount
Summary_communication_tax_type	Summary communication tax type	0:none 1:sct 2:5035 nambered by the low of sct	Summary_communication_tax_type
Tax			Tax
Title			Title
Tool	Tool		Tool
Track_type	Track type		Track_type
Uni_vid	Out of use		Uni_vid
Uniform_resource_locater	Product official url		Uniform_resource_locater
Unit1_coefficient	Unit1 coefficient		Unit1_coefficient
Unit1_height	Unit height (mm)		Unit1_height
Unit1_length	Unit length (mm)		Unit1_length

Unit1_name	Unit name		Unit1_name
Unit1_tare	Unit1 tare		Unit1_tare
Unit1_weight	Unit net weight (kg)		Unit1_weight
Unit1_width	Unit width (mm)		Unit1_width
Unit2_coefficient	Unit coefficient		Unit2_coefficient
Unit2_height	Unit height (mm)		Unit2_height
Unit2_length	Unit length (mm)		Unit2_length
Unit2_name	Unit name		Unit2_name
Unit2_tare			Unit2_tare
Unit2_weight	Unit net weight (kg)		Unit2_weight
Unit2_width	Unit width (mm)		Unit2_width
Unit3_coefficient			Unit3_coefficient
Unit3_height	Unit height (mm)		Unit3_height
Unit3_length	Unit length (mm)		Unit3_length
Unit3_name	Unit name		Unit3_name
Unit3_tare			Unit3_tare
Unit3_weight	Unit net weight (kg)		Unit3_weight

Unit3_width	Unit width (mm)		Unit3_width
Unit4_coefficient	Unit coefficient		Unit4_coefficient
Unit4_height	Unit height (mm)		Unit4_height
Unit4_length	Unit length (mm)		Unit4_length
Unit4_name	Unit name		Unit4_name
Unit4_tara			Unit4_tara
Unit4_weight	Unit net weight (kg)		Unit4_weight
Unit4_width	Unit width (mm)		Unit4_width
Value added to tax	Vat		Value added to tax
Warehouse_code	Warehouse address		Warehouse_code
Warranty_period	The predicted warranty period		Warranty_period
Warranty_period_type	Type of warranty period	0: month 1: day 2: year	Warranty_period_type
Wholesale_rate	Wholesale vat rate		Wholesale_rate
Will_sent_to_web	Will it be sent to the web		Will_sent_to_web
Yield	Yield		Yield, output

Z_report	Z report?		Z_report
Zero_paid_cost_sales_acc_code	Zero paid cost of sales acc. Code		Zero_paid_cost_sales_acc_code