Fast and Efficient Conflict Identification and Resolution in Huge Streaming Data

S. Charles Britto
Research Scholar
Bharathiyar University
Coimbatore, TN, India

S. P. Victor
Associate Professor (CS)
St. Xavier's College
Palayamkottai, TN, India

ABSTRACT
Increased data generation has led to an increase in the availability of rich information online. However, complications occur in the form of heterogeneity in the data storage. In order to have complete information, all the data sources must be utilized. Hence a data integration mechanism is required. However, integrating heterogeneous data leads to conflicting data in the system. This paper presents a fast and efficient mechanism to identify and resolve conflicts on huge streaming data using Spark. A wrapper based query formulation module constructs queries depending on the underlying data sources. The retrieved data is converted to a structured format and similarity between the data is identified, followed by distributed conflict identification and resolution. Experiments were conducted on streaming data. Effective conflict detections and a speed up from ~589 seconds to 10 seconds was achieved.

Keywords
Conflict identification; conflict resolution; Spark; Streaming Data; Wrappers

1. INTRODUCTION
Data integration has become one of the major requirements for any knowledge extraction process since the beginning of the automation of systems. It could be observed that the data generated by systems tend to be in varied formats [1]. This is due to the ease of usage and storage constraints. Accessing or mining such data in isolation is quite simple, as the process can be fine-tuned to act directly on the type of data. However, it has been identified that integrating similar data and then accessing it will provide enhanced knowledge. The crucial stage is the integration mechanism itself. Since the data tends to be heterogeneous, a simple combination technique is not sufficient [2]. Further, extracting data from varied data sources relating to a single user query requires the use of varied querying techniques. Due to the usage of several data sources, the results have high probabilities of containing duplicates and conflicts [3]. Identifying them and eliminating them would provide a huge reduction in the storage requirements. This paper deals with techniques to retrieve data from heterogeneous data sources and combining them to eliminate duplicates and conflicts.

2. RELATED WORKS
Although data integration is a recent technique, several contributions were proposed in this area. Due to the generic nature of this technique, domain based integration and conflict resolution mechanisms could also be observed.

An ontology based technique to resolve semantic conflicts is presented by Han et al. in [4]. Semantic conflict is caused by the varied representations of a single context in heterogeneous systems. This prevents information integration from achieving semantic coherence. The major and mostly used solution for coherence mismatch is ontology based techniques. The technique presented in [4] presents an ontology based schema mapping technique to eliminate sematic conflicts. A similar semantic aware data integration technique for heterogeneous data sources is presented by Leida et al. in [10]. This technique proposes a mapping model, which is an ontology. This makes the mapping model machine understandable. It also utilizes the concepts of semantic join and semantic identifiers to perform semantic data fusion on unrelated data sources. Other similar conflict identification and resolution techniques include [18,19,20].

Object orientation approaches are also used as baselines during the process of data integration. An adaptive approach to heterogeneous data integration using object oriented techniques is presented by Liu et al. in [5]. This paper presents a Distributed Interoperable Object Model (DIOM). DIOM promotes an adaptive approach for interoperation between data sources [6,7]. The major aim of DIOM is to improve the robustness and scalability of services. It uses the DIOM Interface Definition Language (DIOM IDL) [8] to describe models. This is based on the ODMG 93 object model [9] and the main idea of this technique is to incorporate scalability, heterogeneity and huge data volume.

A reverse cleaning technique optimized for integrating heterogeneous multimedia data is presented by Chen et al. in [11]. This technique uses a two phase quality improvement algorithm for data integration. The first phase uses threads to refresh the data as and when modifications are done on the data source. This keeps the data up to date and hence improves the data quality. The second phase performs data reverse cleaning such that the original data is easily accessible even after the cleaning process. Accuracy assessment is performed by using the data accuracy assessment algorithm.

A specialized data integration scheme that operates on Entity-Relationship based data is presented by Lee et al. in [12]. This technique integrates ER schemas and focuses on the resolution of structural conflicts. The major resolution focus is on resolving conflicts between an entity type in one schema and attribute in another schema. It has been observed that performing this process automatically resolves other structural conflicts.

Specific application based data integration techniques include [13,14,22,23]. These techniques are focused on integrating data for knowledge processing in specific application scenarios. Several data structures [15,16,17] were proposed for the integrating schema. They propose flexible structures to efficiently store, retrieve and process data. A deep kernel dimensionality reduction technique was proposed in [21]. This technique has a scalability advantage, hence has the ability to adapt depending on the data size.

3. FAST AND EFFICIENT CONFLICT IDENTIFICATION AND RESOLUTION IN HUGE STREAMING DATA

Identifying and resolving conflicts is a major part of any data integration system utilizing several homogeneous or heterogeneous data sources. Requirements for such systems include faster processing on streamed data in-order to enable quick access for the user. This approach for conflict identification and resolution operates on huge streaming data to provide quick and accurate results (Figure 1).

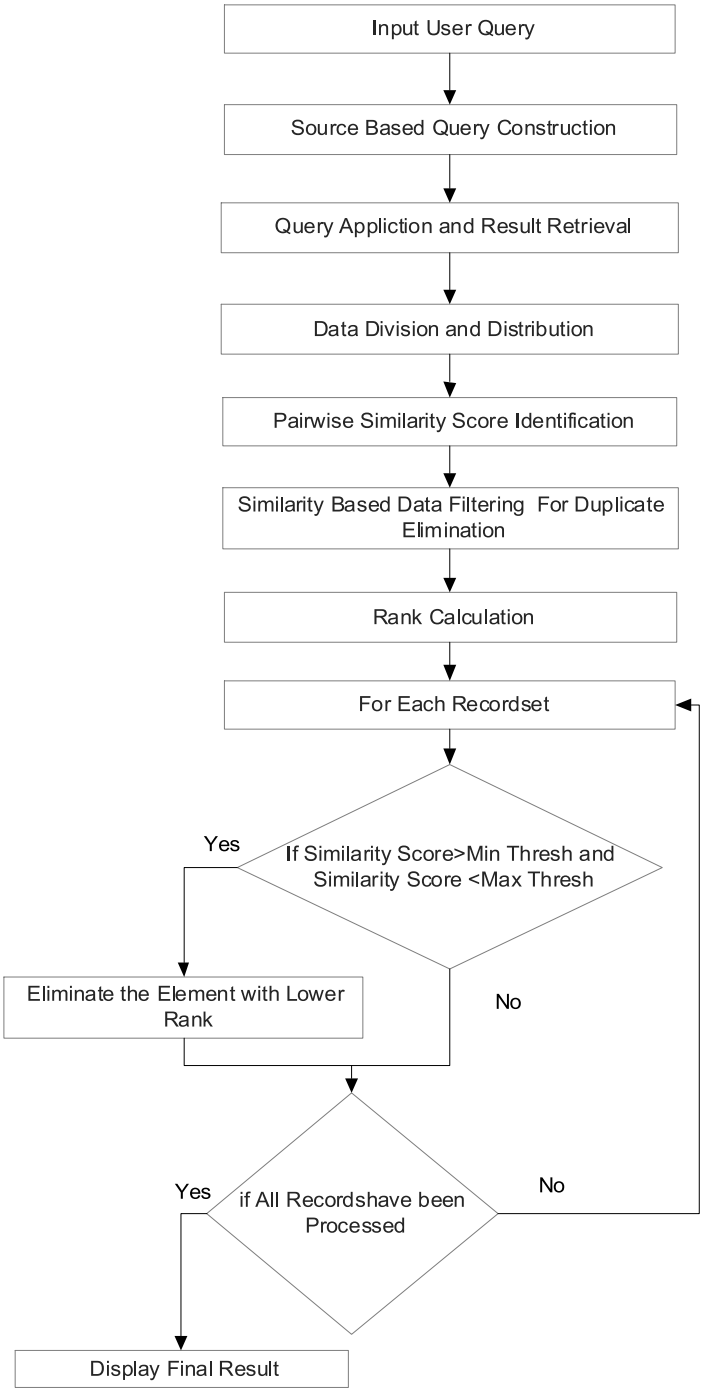

Figure 1: Conflict Identification and Resolution in Huge Streaming Data

The entire process is broadly divided into three major phases; query construction and application, result formulation and similarity identification and rank based filtration to remove duplicates and conflicts.

3.1 Query Construction and Application

User query is obtained in a raw format requiring only the mandatory identifying component representing their requirements. This method of user input reduces the complexity for the user component and provides the raw framework on which the query could be built upon. As the

data integration process requires retrieving and aggregating data from several heterogeneous data sources, obtaining the query in its most basic form proves to be a huge advantage.

Source based query is constructed on the basis of the data sources being used. This phase uses the technique of wrapper based query building. A wrapper builds the query on the basis of its underlying data source. A single wrapper is used for every data source being used. The number of wrappers used in the application depends upon the number or type of data sources that are to be queried for results (Figure 2). A data source can be heterogeneous varying from a database, a web page, an XML document or even an unstructured text document.

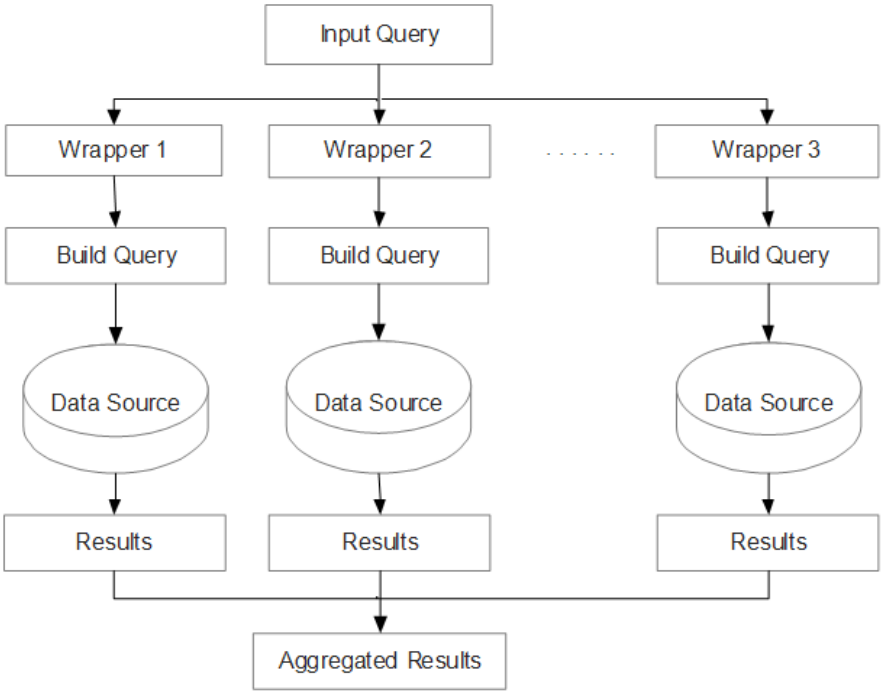

Figure 2: Query Construction and Application

The constructed query is applied on the corresponding data sources and the results are retrieved. The results retrieved from the heterogeneous data sources also vary in their formats. They cannot be directly processed. Converting the retrieved results to a unified format is necessary, which is performed by the next phase.

3.2 Result Formulation and Similarity Identification

The result formulation phase aggregates all the results and converts them to a unified format. The formatted results contain the retrieved results and also the metadata associated with the results. The metadata includes data source, timestamp associated with the data, query association level, title etc. The data is associated with several other related information, hence JSON is used as the base format. Some data sources might not contain all the required metadata fields. Such fields are added with NULL values and during processing, default values replaces the NULL strings.

The structured records are then paired in combinations of two and similarity levels between them are identified. A directional similarity score referred to as sim_d *(Ti, Tj)* is computed between two texts *Ti* to *Tj* using the following equation

$$sim_d(T_i, T_j) = \frac{\sum_{w_i} maxSim(w_i, T_j).idf(w_i)}{\sum_{w_i} idf(w_i)}$$

Therefore, for each word *Wi* in *Ti*, its best-matching counterpart in *Tj* is required (*maxSim(Wi, Tj)).* The similarity scores of all these matches are summed up and weighted according to their inverse document frequency, and then they are normalized.

The final document-level similarity is the average of applying this strategy in both directions, from *Ti* to *Tj* and vice versa

$$sim(T_i, T_j) = \frac{1}{2}(sim_d(T_i, T_j) + sim_d(T_J, T_i))$$

Every pair (*Ti, Tj*) is examined and in pairs exhibiting similarity scores > 0.9, the document *Tj* is eliminated from the corpus. This process is carried out for each similarity pair and the final corpus contains elements that are eliminated of all the duplicates.

3.3 Rank based Conflict Resolution

The corpus is now void of duplicates, however the presence of conflicting documents cannot be avoided when integrating data from several sources. Conflicts usually occur when a context presented by a document provides a contradicting view of the same context in another document. In real time, this occurrence is inevitable. All documents with similarity > 0.4 are considered to be in conflict. All other documents have very low similarity, hence are considered to have contents that have no similarities.

The documents are initially ranked to identify their importance levels in the corpus. Rank of a document is identified using the weighted sum method. It has been discussed in the earlier section that the structured JSON format contains metadata of the retrieved content. Every

property is provided an importance value called its weight (w). Weights are assigned by the user depending on their priority. The actual value of the property is multiplied with its weight and are added to obtain the rank of the data.

$$R_d = \sum_{i=1}^{n} w_i \times v_i$$

Where Rd is the rank of the document with n properties, *wi* and *vi* are the weight and value of the property *i* respectively.

If two documents (*Ti,Tj)* were identified to be in conflict, the document with higher rank is retained and the one with a lower rank is eliminated from the corpus. However this is not carried out as a rule of thumb. If the two documents have very close ranks, both considered of almost equal importance, hence are retained. The closeness level is defined by the threshold ($Thresh_{cl}$) set by the user. The resultant documents are considered to correspond to the user's query and hence are provided to the user.

4. RESULT AND DISCUSSION

The process of conflict identification and resolution is performed by retrieving results using API's and also using the available data sources, and is coded in Python and executed on PySpark. The experimental setup consists of Hadoop 1 deployed in a cluster containing 10 data nodes, 1 name node, 1 secondary name node and 1 client. Results obtained from these varied data sources are passed to the wrappers and the final dictionary is created for processing. The process of identifying conflicts is carried out on heterogeneous data, hence have a variety of data. This property requires the use of an architecture that can operate on structured, as well as unstructured data, hence HDFS is used as the base for storing files. Since the operations are carried out on data obtained from API's, the operations tend to require stream processing, hence Spark is used for operating on the data. Queries were passed through the APIs and the results were passed to the pySpark code for conflict resolution. The entire process is carried out in a single pipeline. Varying queries were passed and 15 different set of executions were carried out.

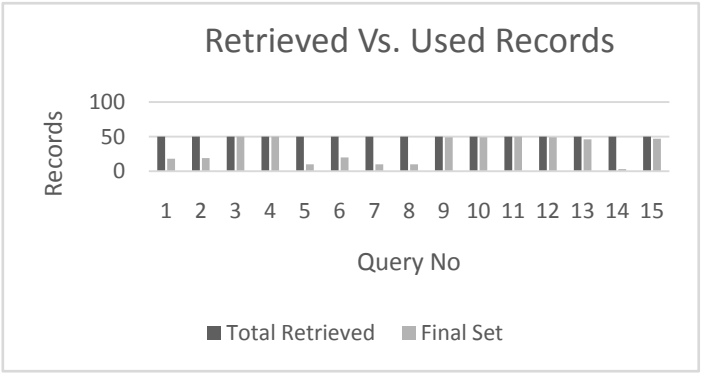

Fig 3: Retrieved Vs. Used Records

The ratio between the number of records retrieved and the number of records that have been finally used is presented in figure 3. An inverse of this process, representing eliminated records are presented in figure 4. Due to the usage of API's, the retrieval levels remained the same (50 records). However it could be noted that the final record count levels went low for some queries, while remained high for others. The higher data levels represent queries for latest information, while the others represent queries on information that are old. This depicts that the algorithm operates on all the types of data, providing only the required eliminations.

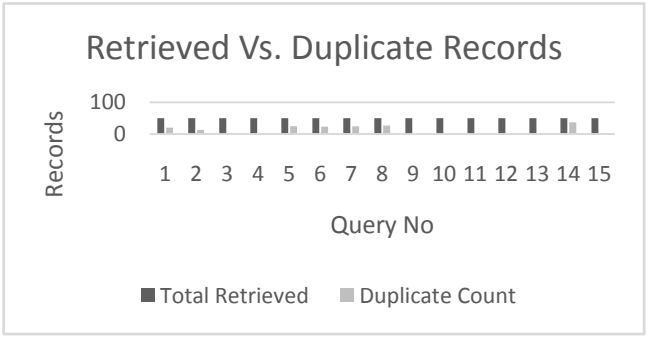

Fig 4: Retrieved Vs. Duplicate Records

Figure 5 represents the elimination levels. The elimination levels are presented in percentages. It could be observed that the elimination levels as high as 95% were obtained on aged data, whereas low elimination levels could also be observed on latest data preserving the information. The low and high fluctuations can be attributed to the recency of the query being used. Recent information tends to contain many articles representing the same entity. Hence it leads to a high probability of occurrence of duplicates, whereas old information tends to get consolidated, hence probability of duplication occurrence tends to get minimized.

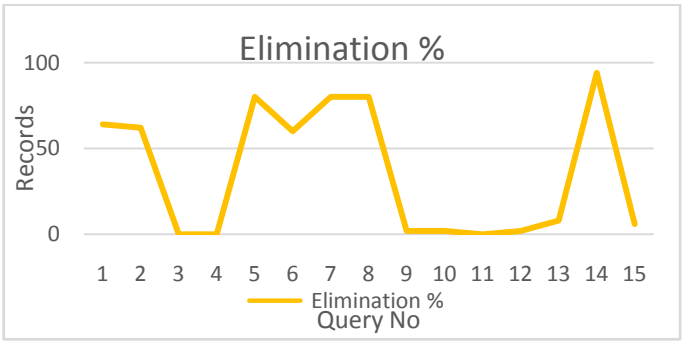

Fig 5: Elimination %

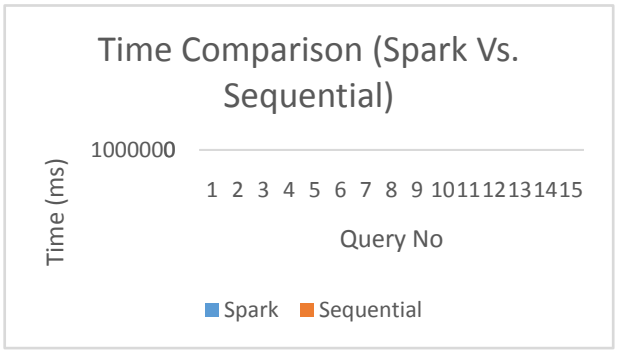

Fig 6: Time Comparison (Spark Vs. Sequential)

A time comparison between Spark based operations and sequential conflict identification and resolution techniques is presented in figure 6. Sequential conflict identification and resolution is carried out by using the python implementation of the architecture, without including the Spark based operations. It could be observed that the sequential operations required huge time, whereas the time taken for Spark based operations is very low. An average time comparison chart is presented in figure 7 along with the data table to better comprehend the time difference. It could be observed that the time taken for sequential python is 589 seconds (approx. 9.8 minutes), while spark based operations requires only 10.9 seconds.

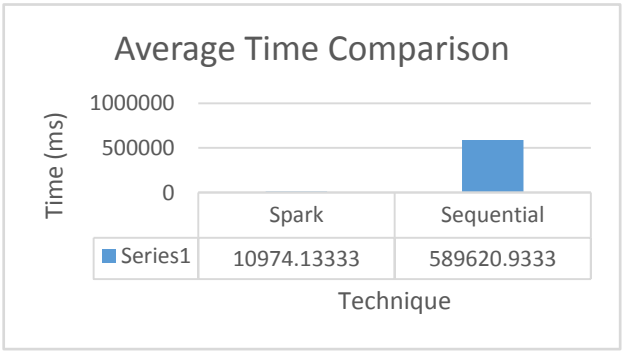

Fig 7: Average Time Comparison

5. CONCLUSION

This paper presents an effective conflict identification and resolution technique using a distributed and stream processing architecture, Spark based on HDFS. It was observed that the Spark architecture performed efficiently in identifying and resolving conflicts with low time latencies. The major advantage of this approach is that wrappers utilized for query formulation are designed as independent components. Hence it is possible to dynamically add or remove wrappers. Hence the proposed architecture has the flexibility to dynamically add or remove data sources. The query provided by the user and the retrieval architecture operates on the content contained in the document. Several other factors such as query context, polarity and several other factors play a crucial role in identifying the appropriate result. The future contributions will be based on identifying and utilizing such factors during the retrieval and the elimination process for more accurate results.

6. REFERENCES

[1] Ohm, J. 2015. Transmission and Storage of Multimedia Data. In Multimedia Signal Coding and Transmission (pp. 491-520). Springer Berlin Heidelberg.

[2] Park, J., 1999. Facilitating interoperability among heterogeneous geographic database systems: a theoretical framework, a prototype system, and evaluation.

[3] Chen, H., Ouyang, Y. and Jiang, W., 2015. An optimized data integration model based on reverse cleaning for heterogeneous multi-media data. Multimedia Tools and Applications, pp.1-16.

[4] Han, L. and Qing-zhong, L., 2004. Ontology based resolution of semantic conflicts in information integration. Wuhan University Journal of Natural Sciences, 9(5), pp.606-610.

[5] Liu, L. and Pu, C., 1997. An adaptive object-oriented approach to integration and access of heterogeneous information sources. Distributed and Parallel Databases, 5(2), pp.167-205.

[6] Wiederhold, G. 1993. Intelligent integration of information, in Proceedings ofACM/SIGMODAnnual Conference on Management of Data.

[7] Wiederhold,G. 1994. Interoperation, mediation, and ontologies, in Proc. Int. Symp. on Fifth Generation Comp Systems, ICOT, Tokyo, Japan, pp. 33–48.

[8] Liu, L. and Pu, C., 1995. Customizable information gathering across heterogeneous information sources. Technical report, Department of Computer Science, University of Alberta.

[9] Cattell, R. et al. 1994.The Object Database Standard: ODMG-93 (Release 1.1). Morgan Kaufmann.

[10] Leida, M., Gusmini, A. and Davies, J., 2013. Semantics-aware data integration for heterogeneous data sources. Journal of Ambient Intelligence and Humanized Computing, 4(4), pp.471-491.

[11] Chen, H., Ouyang, Y. and Jiang, W., 2015. An optimized data integration model based on reverse cleaning for heterogeneous multi-media data. Multimedia Tools and Applications, pp.1-16.

[12] Lee, M.L. and Ling, T.W., 2003. A methodology for structural conflict resolution in the integration of entity-relationship schemas. Knowledge and information systems, 5(2), pp.225-247.

[13] Sandhya, H. and Roy, M.M., 2016. Data Integration of Heterogeneous Data Sources Using QR Decomposition. In Intelligent Systems Technologies and Applications (pp. 333-344). Springer International Publishing.

[14] Bao, J.M., Hu, T.T., Pan, L., Xu, H. and Hu, H.F., 2014, December. Heterogeneous Data Integration and Fusion System Based on Metadata Conflict Algorithms in USPIOT. In Wireless Communication and Sensor Network (WCSN), 2014 International Conference on (pp. 95-100). IEEE.

[15] Isnard, E., Perez, E., Bercaru, R., Galatescu, A., Florian, V., Conescu, D., Costea, L. and Stanciu, A., 2004. Integration and maintenance of heterogeneous applications and data structures. In Advances in Information Systems (pp. 181-191). Springer Berlin Heidelberg.

[16] Chromiak, M. and Stencel, K., 2012. The linkup data structure for heterogeneous data integration platform. In Future Generation Information Technology (pp. 263-274). Springer Berlin Heidelberg.

[17] Comito, C. and Talia, D., 2006. Grid data integration based on schema mapping. In Applied Parallel Computing. State of the Art in Scientific Computing (pp. 319-328). Springer Berlin Heidelberg.

[18] Boufares, F. and Ben Salem, A., 2012, March. Heterogeneous data-integration and data quality: Overview of conflicts. In Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), 2012 6th International Conference on (pp. 867-874). IEEE.

[19] Mirza, G.A., 2015, December. Null Value Conflict: Formal Definition and Resolution. In 2015 13th International Conference on Frontiers of Information Technology (FIT) (pp. 132-137). IEEE.

[20] Chirathamjaree, C. and Mukviboonchai, S., 2002, October. The mediated integration architecture for heterogeneous data integration. In TENCON'02. Proceedings. 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering (Vol. 1, pp. 77-80). IEEE.

[21] Sokolovska, N., Clément, K. and Zucker, J.D., 2016. Deep kernel dimensionality reduction for scalable data integration. International Journal of Approximate Reasoning, 74, pp.121-132.

[22] Calvanese, D., Liuzzo, P., Mosca, A., Remesal, J., Rezk, M. and Rull, G., 2016. Ontology-based data integration in EPNet: Production and distribution of food during the Roman Empire. Engineering Applications of Artificial Intelligence, 51, pp.212-229.

[23] Laraichi, S., Hammani, A. and Bouignane, A., 2016. Data Integration As The Key To Building A Decision Support System For Groundwater Management: Case Of Saiss Aquifers, Morocco. Groundwater for Sustainable Development.