Data Quality Alerting Model for Big Data Analytics

Eliza Gyulgyulyan, Julien Aligon, Franck Ravat, Hrachya Astsatryan

Data Quality Alerting Model for Big Data Analytics

Eliza Gyulgyulyan[1,2], Julien Aligon[2], Franck Ravat[2], Hrachya Astsatryan[2]

[1] IIAP, National Academy of Science of Republic of Armenia,
1 Paruyr Sevak str., Yerevan, Armenia
[2] IRIT-CNRS (UMR 5505) – Université Toulouse 1 Capitole,
2 Rue du Doyen Gabriel Marty, 31042 Toulouse Cedex 9, France
firName.lastName@irit.fr

Abstract. During Big Data analytics, correcting all the problems of large, heterogeneous and swift data, in a reasonable time, is a challenge and a costly process. Therefore, organizations are confronted with performing analysis on massive data, potentially of poor quality. This context is the starting point of our current research: how to identify data quality issues and how to notify users without solving these quality issues in advance? To this end, we propose a quality model, as the main component of an alert system, which allow to inform users about data quality issues, during their analysis. This paper discusses about the conceptual and implementation frameworks of the quality model, as well as examples of usage.

Keywords: Quality Model, Data Quality, Big Data Analytics.

1 Introduction

Big Data analytics has undeniable vast importance as it has been absorbed into almost all aspects of scientific or industrial activities. In today's digital world an enormous amount of data is collected, categorized, stored for further analysis, with increasing speed. Decision-makers or knowledge-workers analyze data coming from different sources in order to make a fast and fair analysis.

The tremendous volume of data changing with high speed, the consideration of Data Quality (DQ) requires a higher amount of time and higher processing resources [1, 2].

Moreover, the context of Big Data induces always more diversity of data sources, data types or data structures. This diversity increase the difficulty of correcting every type of quality issue [1]. If handling traditional DQ problems is a challenge, then handling them, facing with the Big Data analytics context, is even more challenging. On the other hand, doing analysis on faulty, poor and untrustworthy data can have considerable and negative consequences for companies. According to [3], quality problems cost to US businesses around 600 billion dollars annually. Thus, performing analysis using rough data and obtaining a reliable result is a real issue in the context of Big Data.

This paper considers analysis in a Big Data context, without using any correction processes, beforehand. Indeed, producing *Value* from analysis results, using a minimum of resources for solving quality problems, is our main motivation. This work can be

viewed as a continuation of the impact study of the classical 5V's over the Big Data *Value* [2]. To this end, our main research questions refer to the three following challenges:

- How is it possible to identify DQ problems during the analysis?
- How the user can know the quality of the data he/she is analyzing?
- Can the user obtain a good analysis result without solving all the quality problems on the data beforehand?

The first solution to face our problematic is to alert about poor quality when an analysis is being done. An alert system is suggested to notify users about the quality problems along their analysis. The system gives the possibility to refer to DQ problems (or part of them) described in [4] during the analysis process only if the problems are relevant for the ongoing analysis . The notion of alerting about the quality problems can make the process of Big Data analysis more accurate as the analysis is being done without data correction and the result can be biased. Moreover, it will prevent companies from spending resources on solving all the quality problems before the analysis.

The aim of this paper is to describe the main components of the alert system, in particular the quality model. We provide the characteristics of this model thanks to a conceptual schema and an implementation of it, illustrated with examples. Referring to the Big Data context, the model is designed considering quality characteristics and data source diversity. Thanks to the quality model, the alert system can offer to the user the opportunity to ask quality questions over the data sources he/she wants to analyze.

The rest of the paper is organized as follows: Section 2 describes the related work of the DQ field, both in a general approach and in the context of Big Data, problems and our suggestions. Section 3 presents our main contribution i.e. the quality model, to be used in our future alert system. This model is defined by a conceptual model and a physical representation of it. The rules of transformation between the conceptual components and the implementation are also given. The implementation is realized in a graph database, in particular using Neo4j[1] database. The last section concludes our paper and gives several perspectives of work.

2 Related work

DQ itself is considered as a measurable notion describing the level of a set of qualitative and quantitative dimensions and metrics describing it. DQ dimensions are widely and differently discussed and described in the literature [1, 5–13]. Most of the articles identifies various DQ dimensions like consistency, accuracy, timeliness, completeness, etc. There are more than 170 dimensions specified for the last 20 years. Some concepts are very close to each other. In the Big Data analytics context, the majority of identified quality dimensions coincide with the traditional DQ dimensions in database field.

In the context if DQ, metrics are classically required for each dimension. A quality metric is a standard of measurement to compute the dimension. A wide used method

[1] https://neo4j.com/developer/get-started/

for choosing appropriate metrics has been suggested in [14], called the Goal-Question-Metric approach. The idea of this approach allows a user to ask a Question, over his data and related to his own quality goal, the answer of which provides the corresponding metric. This approach is used in the quality meta-model of the ESPRIT project [15] and quality assessment meta-model of the QUADRIS project [7].

In the Big Data context, the majority of possible data quality dimensions coincide with the traditional DQ dimensions. such as *consistency* [1, 8, 9, 16–18], *uniqueness* [6, 8, 10, 18], *accuracy* [1, 8, 16–19], *completeness* [1, 16–19], *timeliness* [3, 5, 6, 11–13]. Still, literature provide discussions of Big Data quality dimensions and metrics [1, 3, 16–21]. When dealing with different information sources, complementary values related to a same entity (for instance, the name of a person is present in one source, and the surname in another one) should also be considered. Thus, considering *uniqueness*, not only purging duplicate values but also merging complementary values is essential [8]. Also, *synchronization* is important to obtain a consistent data [1]. *Interpretability* [6, 9, 12, 18] delivers the notion of extracting a good *Value* from Big Data. *Data trustworthiness* is one of the major attributes of data Veracity, which is one of the 7 V's of Big Data related to quality [2]. It is defined by a number of factors including data origin, collection and processing methods, including trusted infrastructure and facility [22]. Reputation and credibility of data source is considered a highly regarded level of trustable data [1, 9, 16, 18, 19].

The general notion of quality in Big Data has also been widely used in the literature. In the context of Big Data, quality is not limited to data: quality dimensions are discussed also for system quality [23, 24] or quality for analysis platform [25]. Moreover quality models for Big Data initiatives are presented in the literature to handle quality issues and deal with the quality evaluation, assessment, and management. A discovery model of quality constraints for lake data has been suggested in [26] and a quality-in-use models (3As and 3Cs) have been suggested in [19, 20]. The latest papers describe quality assessment in big data projects based on ISO/IEC standards. However, it considers the quality-in-use approach which assess the quality of data for Big Data projects providing the appropriate data quality for Big Data analysis. Nevertheless, [16] gives a good literature review about the notion of quality in Big Data context. However, these works consider data preprocessing as quality provider where the analysis are done on already processed data. In other words, in the literature, analyzing data quality only implies on pre-processing of Big Data analytics [16]. Also, the quality models reflected in the literature mainly consider data quality on a single source. Another flaw is that literature does not take into account user preferences sufficiently.

We propose a more complete model (compared to the literature), which is applicable on multiple data sources of even different types. In addition, we believe there are organizations that cannot afford (in terms of time and resources) [3] data quality corrections before the analysis process and consider all of the existing quality problems in their data. These problems are described by the level of quality dimensions e.g. how consistent data is. Thus, we ambition to alert about data quality problems during the analysis stage without any preprocessing. Moreover, the alert system takes into account user preferences. It should directly give an opportunity to a user interact with it and the quality model via an interface.

3 The Quality Model

In the following figure (Fig. 1), we present a roadmap of the alert system describing the interactions of the alert system with the users and the Big Data sources. The user can either directly query the data through the alert system or interact with the quality model through a user interface to choose the quality questions. This section is dedicated to the description of the quality model.

3.1 Conceptual model

The quality model of our alert system represents the quality assessment of a data source in the context of Big Data Analytics. It is based on quality dimensions and metrics described in Section 2. The user is able to use this model through quality questions related to the analysis he/she performs. The main concepts of the quality model and the interactions between them are illustrated in Fig. 2. The assessment of a metric on a data source and/or attribute is done thanks to a predefined measurement method. A central concept of this model is the consideration of "Quality Question". This concept is seen as a "negotiator" between the user and the quality assessment of the model.

Fig. 1. The roadmap of the alert system.

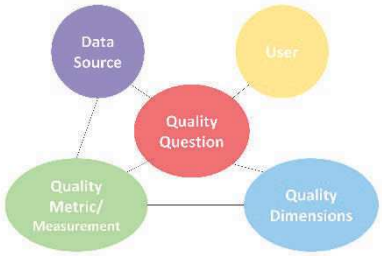

Fig. 2. Main entities of quality model.

The conceptual model is represented in Fig. 3. In order to be independent of any implementation considerations, we model our solution with a class diagram. The description of each class is detailed below:

Quality question: This class expresses the user requirements about the quality assessment. The user chooses the dimensions and/or the metrics he wants to be alerted about from a predefined list (this list can be extended later due to the Neo4J functionalities such as adding nodes and relationships easily), according to his/her analysis. This class refers to data sources and attributes. If the user is unable to express particular dimensions or metrics, the model considers all the dimensions and metrics by default. The latest option is not preferable if the user is sophisticated enough in the quality domain.

Data source and Attribute: These two classes describe the Big Data substance on which the user performs analysis. The user can perform an analysis on one or several attributes in order to have a specific quality assessment or consider a global analysis by

analyzing the whole data source. Here, different types of data sources and attribute formats are supported by the model. Our quality model should be able to operate on the data sources regardless of their type, meaning all types of data sources need to be supported by the model.

Source set: When querying multiple data sources and/or attributes, the user queries a set of them. Here, data structure should be considered. In a single source set the data structure need to be uniform.

User: This class indicates who performs the analysis and who, if able, chooses quality questions on data source, attribute, and/or source set. This class is essential as it defines user rights in terms of "role". In other words, this class can personalize the data quality information, of a same data sources, to different users. The "role" property considers two types of users: 1) a user not sophisticated enough in the domain of quality (for instance, a decision-maker or analyst): this user does not choose quality questions but the system chooses instead by default, 2) a user sophisticated enough to choose quality questions, and set quality limit as a threshold.

Quality dimension: This class considers the dimensions listed in Section 3. The dimension is considered as a view of quality assessment for the user. When checking the quality, the user is able to select one or several quality dimensions in his Quality question. The relationships between dimensions (such as improving completeness can have negative impact on uniqueness.) should also be reflected by the model, for suggesting quality improvements by the system. It has been already noted, that discovering and analyzing relationships of quality dimensions plays a significant role when rationing an analysis process [27], and even a dependency discovery model has been suggested in [28]. Particularly, negative (inverse) relationships are important to be considered in our model. These relationships will help user to make a decision about the improvement of a particular dimension as he/she will also be informed about possible deviations on other dimensions in case of modification. The direct relationships should not be considered as there is nothing to alert about. It is even better that one more dimension is going to be improved in case of the improvement of the detected dimension.

Quality metric: This class refines the Quality question and is a way of quality computation addressing the Quality Dimension. For instance, "NullValues" is a metric of the dimension "completeness" [6]. In case where this metric is specified in the Quality question, the model should check the level of "null values", over the data, to alert about completeness. If no metric is specified in the Quality question (only dimensions are expressed), the model considers all the metrics of that dimensions.

Measurement method: This class defines a quality formula which measures the problem of quality for a specific metric. The measurement method represents the implementation of the metric computation (a formula, an algorithm) on data source, attribute, or source set. In the literature there are several algorithms already developed for quality dimension evaluation such as [17], which can be a base for this class to compute the level of a quality metric. For instance, to compute the dimension of "completeness", the model needs to compute the metric "Number of NullValues" using the "CheckNULL" function. Thus, the formula "[(1-Number of not null values)/total number of values]" is calculated to alert about the "completeness" dimension. The complete set of

measurement methods (such as CheckNULL, CheckRule, CheckReferential, Aggregation, LookUp, Count, Ratio, Max, Min, etc. [7] need to be implemented.

Quality Limit: This class considers a limit value as a threshold for which the system alerts the user. This value is specific for each measurement method and can be entered by the user. Of course, this value must depend of the measurement domain. For instance, if the domain of values for the measurement "null values" is between [0,1] (0-worst case, 1-best case), the user could specify a value of 0.9, and the system alerts only if the result is out of this limit. If the user does not select a limit, the model considers the limit as the lowest value of the domain for the concerned quality problem.

3.2 Quality model example

We illustrate the quality model of the previous section, considering the classical example of Sales, when a user wants to analyze product sales. Let us consider the object diagram of Fig. 4. Without going deep into the analysis process, we consider the scenario when the user is sophisticated enough to handle the quality model by choosing particular quality questions. In the suggested example, a knowledge-worker Arsen (U1:User) analyzing sales of a product. The data are stored in a local server via a HDFS (Hadoop Distributed File System). He performs the analysis on the attribute "productSale" (A1:Attribute). Arsen prefers to check over the completeness and consistency dimensions to be sure that the attribute he is analyzing has no quality problem. Besides, Arsen understands that if there will be duplicates on the data, this number may be exaggerated. That is why he also decides to check over the uniqueness of the attribute "productID" (A2:Attribute). Thus, during his analysis, Arsen chooses the quality questions (Q1) and (Q2). Please note, in the object diagram, everything concerning the completeness part is colored pink, consistency – yellow, and uniqueness - green:

- (Q1) considers the completeness and consistency dimensions. Because Arsen didn't specified metrics for these dimensions, the model considers all the metrics for each of these dimensions (of course the full list of dimensions and metrics is stored in the model, beforehand). Thus, the corresponding metrics for completeness are (Com1), (Com2), and for consistency (Con1). For each quality metric two measurement methods are predefined. The method (MCom1) only counts the number of null values and returns it as a result, whereas the other method (MCom2) checks the number of "null values" and computes a ratio by giving the proportion of "not null values". Thus, when alerting the alert system presents both results to Arsen. Then, he knows that there are 500 null values in the attribute "productSale" and the proportion of "not null values" is 0.5. Now, let us discuss (Com2). In this case the metric considers the whole data source and not a specific attribute. Arsen selects quality limit [0.75;1]. This means that Arsen does not mind to analyze the sales even if the amount of data is 75% of the needed data amount (e.g. usually X amount of data is needed but this time 0.75*X is enough). That is why, the system will not alert, though the measurement method (MCom3) returns a result of 0.82.

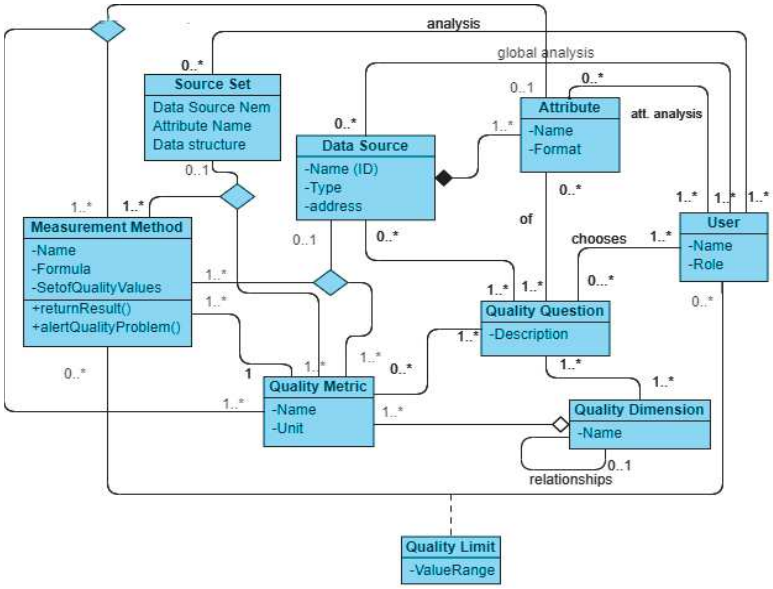

Fig. 3. Conceptual model of the quality model.

– (Q2) Arsen mentions that only the duplicates need to be checked. This prevents the model to check over other quality metrics of the dimension "uniqueness" that are not specified. And thus the system will only alert about the number of duplicate values. Also, when suggesting an improvement of completeness, the system will consider the negative relationship (see Section 4.1) between the completeness and uniqueness and will notify the user about it. This alert message should also contain an information about the level by which the uniqueness will suffer if the completeness is improved.

3.3 Physical representation of the quality model

We intend to implement the quality model through a NoSQL database. Due to the fact that different concepts of the model are interrelated, we choose a graph database. For example, a same metric may be shared by several measurement methods and may be

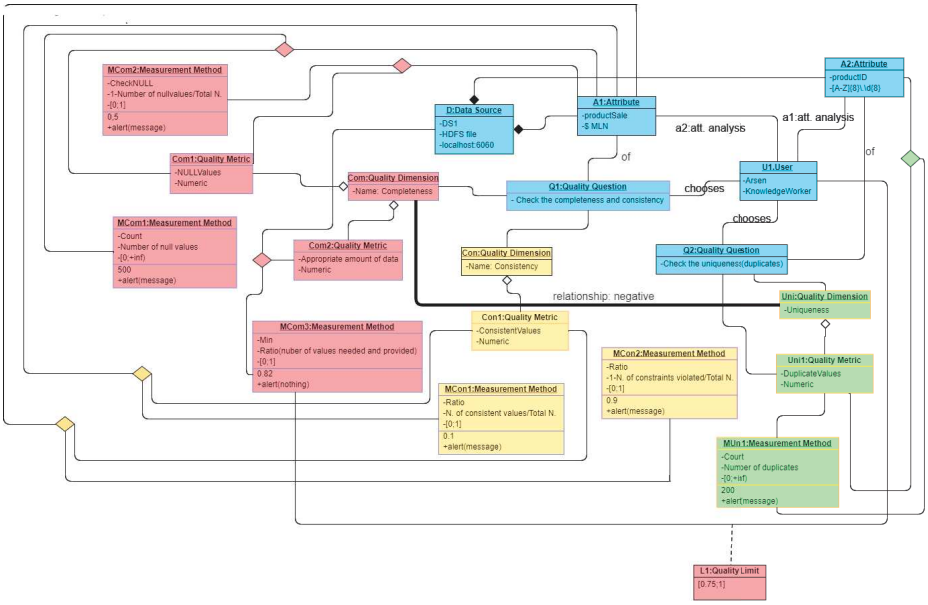

Fig. 4. The object diagram of the example when user analyzes sales.

applied on several data sources. Thus, these concepts can easily be queried using a graph query language.

Various definitions of a graph exist in the literature. By combining the common definition of a graph presented in [29] and the basics of Neo4J the graph can be defined as follows: the graph G is an ordered pair of (N(G), E(G)) consisting of a nonempty set N(G) of nodes and a set E(G) of edges, which represents the relationship between an unordered pair of (not necessary distinct) nodes of G. Thus a graph database considers as equally important the relationships as the data itself.

The delightful part of the graph database is also the possibility to extend the existing model, by adding new nodes (e.g. set of quality questions) and/or relationships continuously, without changing the complexity of the queries. While classical relational databases compute relationships at query time through expensive JOIN operations, a graph database stores connections alongside the data in the model. This is a good way for our quality model to be enhanced and supplemented over the time. There will be a possibility to add new dimensions, metrics, relationships and data source types thanks to our model. Also, retrieving nodes and relationships in a graph database is an efficient, constant-time operation, which traverse numerous of connections per second per core.

We use Neo4J community version (see Fig. 5) as the graph database for our model, using the query language - Cypher. The main concepts of Neo4J are labels, nodes, properties and relationships. In order to implement our conceptual model, translation rules are required. Methods for translating conceptual schema of data model exist in the literature [30, 31]. The translation of our class diagram in Neo4J is done by considering the mapping from [31]. From the global point of view, the classes of class diagram are

"label" node in Neo4J, the attributes are "properties", and the associations are "relationships".

Considering the translation the data is imported into Neo4J database. An extract of the database may be seen in Figure 5. This schema presents the labels of the quality model (not the values). All the labels from Figure 5 can be linked with the classes of Figure 3. We present a similar case to the example from Section 4.1 as a graph (see Figure 6). In this case a user Arsen decides to check only over completeness and consistency dimensions of a data source without mentioning any metric in the quality question. This time Arsen wants, to be sure that the data source has no problem with these two quality dimensions as they are the most commonly discussed quality dimensions in the literature.

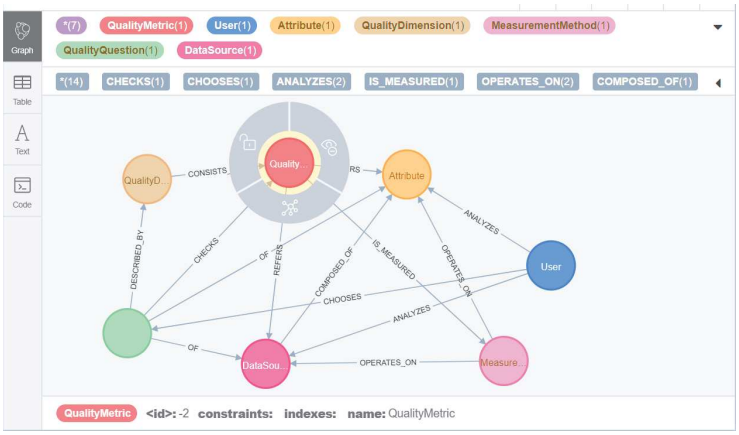

Fig. 5. The quality model implemented into a database schema in Neo4J

The Fig. 6 shows the result of the following Cypher query in Neo4J:

```
MATCH (n:QualityQuestion)<-[:CHOOSES]-(u:User)-[:ANALYZES]-
>(d:DataSource)WHERE n.description=~'Check over Dim. Complete-
ness.*' OR n.description=~'Check over Dim. Cosistency.*'OPTIONAL
MATCH (n)-[:DESCRIBED_BY]->(q:QualityDimension)-[:CONSISTS_OF]-
>(m:QualityMetric)OPTIONAL MATCH (m)-[:IS_MEASURED]-
>(mm:MeasurementMethod)RETURN n, u, d,q,m,mm
```

Because there is no information about the metrics in quality question, all the metrics of the chosen dimensions are considered. Thus, as completeness has three metrics and consistency has one metric, the model considers four quality questions for each metric (despite Arsen chooses only one quality question, the model needs to consider all of the possible quality questions containing the metrics). For each metric the appropriate measurement methods are identified. At the top of the Fig. 6, all the nodes "label" can be seen (at the bottom, the properties of the selected node are visualized).

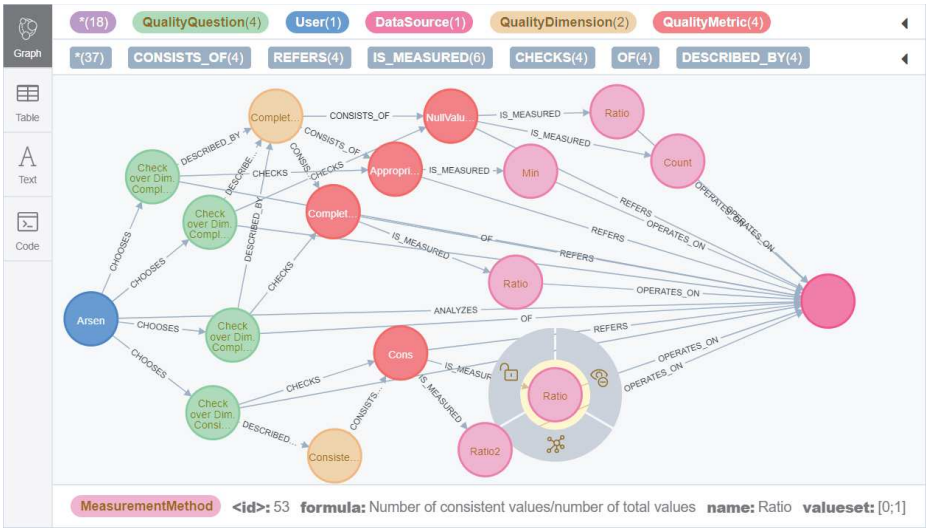

Fig. 6. quality model example in Neo4J

4 Conclusion

In order to avoid solving all quality problems in the context of Big Data analytics, this paper suggests alerting about quality issues when analyzing data directly (without any data quality preprocessing). Thus, an alert system can be considered as a good candidate. To this end, we propose a quality model as the main part of this system. The model allows users to control the quality of their data during the analysis process and be informed about the problems on it. Thanks to the quality metrics and their measured values, the system can be able to alert about poor data quality, according to user requirements. This solution relieves the load of data correction before the analysis and consider them during the analysis. Then, it is up to a user to decide whether the problems need to be improved or not. A Neo4J implementation of quality model is presented with a query example.

The future work includes the definition of new quality dimensions dedicated to the data analysis step [32]. In particular, these dimensions should be able to alert a user about a trustful/untruthful analysis. The integration of relationships between the quality dimensions should enhance the capabilities of it. The system should also alert about the quality of analysis based on the alerted data quality by considering the relationships between the dimensions (data quality dimensions and analysis quality dimensions). Then, the next step will be to design and implement the complete alert system. A long term perspective is also to support and solve automatically the quality problems that are detected (under the supervision of the user, of course).

References

1. Cai, L., Zhu, Y.: The Challenges of Data Quality and Data Quality Assessment in the Big Data Era. Data Sci. J. 14, 2 (2015).
2. Khan, M.A., Uddin, M.F., Gupta, N.: Seven V's of Big Data understanding Big Data to extract value. In: Proceedings of the 2014 Zone 1 Conference of the American Society for Engineering Education. pp. 1–5. IEEE, Bridgeport, CT, USA (2014).
3. Saha, B., Srivastava, D.: Data quality: The other face of Big Data. In: 2014 IEEE 30th International Conference on Data Engineering. pp. 1294–1297 (2014).
4. Erl, T., Khattak, W., Buhler, P.: Big Data Analytics Lifecycle. In: Big Data Fundamentals. pp. 65–87. Arcitura Education Inc., United States (2016).
5. Lee, Y.W., Pipino, L.L., Funk, J.D., Wang, R.Y.: Journey to Data Quality. The MIT Press (2006).
6. Batini, C., Cappiello, C., Francalanci, C., Maurino, A.: Methodologies for Data Quality Assessment and Improvement. (2009).
7. Berti-Équille, L., Comyn-Wattiau, I., Cosquer, M., Kedad, Z., Nugier, S., Peralta, V., Si-Said Cherfi, S., Thion-Goasdoué, V.: Assessment and analysis of information quality: a multidimensional model and case studies. Int. J. Inf. Qual. 2, 300–323 (2011).
8. Rahm, E., Hai Do, H.: Data Cleaning: Problems and Current Approaches. IEEE Data Eng Bull. 23, 3–13 (2000).
9. Wang, R.Y., Strong, D.M.: Beyond Accuracy: What Data Quality Means to Data Consumers. J Manag. Inf. Syst. 12, 5–33 (1996).
10. Akoka, J., Berti-Equille, L., Boucelma, O., Bouzeghoub, M., Wattiau, I., Cosquer, M.: A framework for quality evaluation in data integration systems. (2007).
11. Pipino, L.L., Lee, Y.W., Wang, R.Y.: Data Quality Assessment. Commun ACM. 45, 211–218 (2002).
12. Hazen, B.T., Boone, C.A., Ezell, J.D., Jones-Farmer, L.A.: Data quality for data science, predictive analytics, and big data in supply chain management: An introduction to the problem and suggestions for research and applications. Int. J. Prod. Econ. 154, 72–80 (2014).
13. Ballou, D.P., Tayi, G.K.: Enhancing Data Quality in Data Warehouse Environments. Commun ACM. 42, 73–78 (1999). https://doi.org/10.1145/291469.291471.
14. Oivo, M., Basili, V.R.: Representing software engineering models: the TAME goal oriented approach. IEEE Trans. Softw. Eng. 18, 886–898 (1992).
15. Jeusfeld, M.A., Quix, C., Jarke, M.: Design and Analysis of Quality Information for Data Warehouses. In: Ling, T.-W., Ram, S., and Li Lee, M. (eds.) Conceptual Modeling – ER '98. pp. 349–362. Springer Berlin Heidelberg (1998).
16. Taleb, I., Serhani, M.A., Dssouli, R.: Big Data Quality: A Survey. In: 2018 IEEE International Congress on Big Data (BigData Congress). pp. 166–173 (2018).
17. Taleb, I., Kassabi, H.T.E., Serhani, M.A., Dssouli, R., Bouhaddioui, C.: Big Data Quality: A Quality Dimensions Evaluation. In: 2016 Intl IEEE Conferences on Ubiquitous Intelligence Computing, Advanced and Trusted Computing, Scalable

Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld). pp. 759–765 (2016).
18. Arolfo, F., Vaisman, A.: Data Quality in a Big Data Context. In: Benczúr, A., Thalheim, B., and Horváth, T. (eds.) Advances in Databases and Information Systems. pp. 159–172. Springer International Publishing, Cham (2018).
19. Caballero, I., Serrano, M., Piattini, M.: A Data Quality in Use Model for Big Data. In: Indulska, M. and Purao, S. (eds.) Advances in Conceptual Modeling. pp. 65–74. Springer International Publishing (2014).
20. Merino, J., Caballero, I., Rivas, B., Serrano, M., Piattini, M.: A Data Quality in Use model for Big Data. Future Gener. Comput. Syst. 63, 123–130 (2016).
21. Sidi, F., Panahy, P.H.S., Affendey, L.S., Jabar, M.A., Ibrahim, H., Mustapha, A.: Data quality: A survey of data quality dimensions. In: 2012 International Conference on Information Retrieval Knowledge Management. pp. 300–304 (2012).
22. Demchenko, Y., Grosso, P., Laat, C. de, Membrey, P.: Addressing big data issues in Scientific Data Infrastructure. In: 2013 International Conference on Collaboration Technologies and Systems (CTS). pp. 48–55 (2013).
23. Choi, S.-J., Park, J.-W., Kim, J.-B., Choi, J.-H.: A Quality Evaluation Model for Distributed Processing Systems of Big Data. J. Digit. Contents Soc. 15, 533–545 (2014).
24. Canalejo, O., Isabel, M.: A Quality Model for Big Data Database Management Systems. (2018).
25. Lee, J.Y.: ISO/IEC 9126 Quality Model-based Assessment Criteria for Measuring the Quality of Big Data Analysis Platform. J. KIISE. 42, 459–467 (2015).
26. Farid, M., Roatis, A., Ilyas, I.F., Hoffmann, H.-F., Chu, X.: CLAMS: Bringing Quality to Data Lakes. In: Proceedings of the 2016 International Conference on Management of Data - SIGMOD '16. pp. 2089–2092. ACM Press, San Francisco, California, USA (2016).
27. Berner, E.S., Kasiraman, R.K., Yu, F., Ray, M.N., Houston, T.K.: Data quality in the outpatient setting: impact on clinical decision support systems. AMIA Annu. Symp. Proc. AMIA Symp. 41–45 (2005).
28. Barone, D., Stella, F., Batini, C.: Dependency Discovery in Data Quality. In: Proceedings of the 22Nd International Conference on Advanced Information Systems Engineering. pp. 53–67. Springer-Verlag, Berlin, Heidelberg (2010).
29. Bondy, J.A., Murty, U.S.R.: Graph theory with applications. (1976).
30. Daniel, G., Sunyé, G., Cabot, J.: UMLtoGraphDB: Mapping Conceptual Schemas to Graph Databases. In: ER 2016 - 35th International Conference on Conceptual Modeling. pp. 430–444. Springer, Gifu, Japan (2016).
31. Delfosse, V., Billen, R., Leclercq, P.: UML as a schema candidate for Graph databases. In: NoSQL Matters 2012. pp. 1-8 (2012).
32. Djedaini, M., Furtado, P., Labroche, N., Marcel, P., Peralta, V.: Benchmarking Exploratory OLAP. In: Nambiar, R. and Poess, M. (eds.) Performance Evaluation and Benchmarking. Traditional - Big Data - Internet of Things. pp. 61–77. Springer International Publishing (2017).