From Document Warehouse to Column-Oriented NoSQL Document Warehouse

Ines Ben Messaoud[1], Refka Ben Ali[2] and Jamel Feki[3]
[1]Laboratory Mir@cl, University of Sfax, Sfax, Tunisia
[2]Institute of Management of Gabès, University of Gabès, Gabès, Tunisia
[3]University of Jeddah, Jeddah, Saudi Arabia

Keywords: NoSQL Document Warehouse, Galaxy Model, Column-oriented Model, Hierarchical Transformation.

Abstract: NoSQL (Not only SQL) gathers recent solutions that differ from the SQL model by a different logic of data representation. It is characterized by its performance and its ability to handle a large amount of data. Due to the absence of a clear approach to implement a Document Warehouse (DocW) under NoSQL model, we propose, in this paper, a set of rules to transform the multidimensional galaxy model of the DocW into the column-oriented NoSQL model. We suggest two types of transformations namely *Simple* and *Hierarchical*. In order to validate our proposed transformation rules, we have used Cassandra as a Column-oriented NoSQL system to implement a DocW for each type of transformation. We used Talend Data Integration tool to load data in the implemented DocWs. We evaluate these two DocWs with two metrics WRL (Write Request Latency) and RRL (Read Request Latency) using a medical collection.

1 INTRODUCTION

Document Warehouse provides an environment to store unstructured data; it is a repository of documents collected from external and internal data sources. It is designed according to the multidimensional model as either a star model (Tseng et al., 2006) (Ben Mefteh et al., 2016) or a galaxy model (Ben Messaoud et al., 2015) (Pujolle et al., 2011). It organizes data for OLAP (On Line Analytical Processing) analysis in order to enable successful business intelligence (Tseng et al., 2006). Its main concerns are: (i) the uniform storage of data, and (ii) the restoration of fragments of texts considered as relevant by the user (Tournier, 2007).

However, the volume of data to analyze reaches critical sizes (Jacobs, 2009) and then becomes difficult to manage with the classical available tools; hence, the need of appropriate storage and processing techniques is raised (Agrawal et al., 2011). In this emerging context of “Big Data”, NoSQL environment appears as an efficient alternative that can provide scalability while maintaining flexibility for an OLAP system. In fact, it allows considering new approaches to implementing a warehouse, particularly multidimensional implementation (Chevalier et al., 2015a).

In order to benefit from the NoSQL technology, we propose in this paper an approach to implementing a NoSQL Document Warehouse (DocW). More precisely, we are interested in the column-oriented NoSQL model. It is the most appropriate model for the warehouse and the multidimensional data structure (Dehdouh et al., 2015). In addition, it allows deploying the warehouse in the cloud and offers high performance.

In this paper, we define a set of transformation rules to convert the multidimensional model of the document warehouse into a column-oriented NoSQL model. We implement these rules using Cassandra NoSQL database; we evaluate the results in terms of the two metrics namely *Write Request Latency* and *Read Request Latency* on a medical collection.

This paper is organized as follows: Section 2 discusses related works that address NoSQL Data Warehouses. Section 3 presents the DocW specifics and its relying multidimensional galaxy model. Then, the column-oriented NoSQL model and our proposed transformation rules to obtain a NoSQL DocW are described in Section 4. Section 5 is dedicated to experiment and evaluation. Finally, Section 6 concludes the paper and addresses future works.

Messaoud, I., Ali, R. and Feki, J.
From Document Warehouse to Column-Oriented NoSQL Document Warehouse.
DOI: 10.5220/0006423500850094
In *Proceedings of the 12th International Conference on Software Technologies (ICSOFT 2017)*, pages 85-94
ISBN: 978-989-758-262-2

2 RELATED WORKS

A DocW is a dedicated technology to store documents issued from internal and external data sources to the organization. These documents are organized for effective analyses in order to enable distilled and fruitful business intelligence (Tseng et al., 2006). A DocW can be modelled in different ways, as a star model (McCabe et al., 2000) (Tseng et al., 2006), or as a galaxy model (Pujolle et al., 2011) (Ben Messaoud et al., 2015). Furthermore, with the increasingly volumes of data, it is becoming impractical or even impossible to load unstructured data into a warehouse. To alleviate this problem, new technologies are currently studied in research laboratories at universities and by software providers. Recently, NoSQL (Not only SQL) emerges as a promising technology to deal with huge volumes in conventional databases (Stonebraker, 2012) and in Big Data applications. Furthermore, (Chandawni, 2016) advocate to use NoSQL in the warehousing domain. In fact, to the best of our knowledge, few literature works treat the data warehouse using NoSQL. Furthermore, there is no literature works those treat the Document Warehouse relying their solutions on NoSQL.

In this section, we study pertinent works where researchers have proposed approaches or rules to implement Data Warehouse using NoSQL, precisely column-oriented and document-oriented models.

Among the well-known works, we cite the paper of (Li, 2010) where the author proposes an approach to transform a relational database into a column-oriented NoSQL database. The author presents a set of rules to conduct this transformation. Yet, the proposed approach does not consider the conceptual level of the Data Warehouse (DW); it considers only the logical level to transform the relational model into a column-oriented model.

In (Dehdouh et al., 2014), the authors have presented a benchmark for columnar NoSQL DW. This benchmark allows generating synthetic data and queries set in order to evaluate the performance of systems and the impacts of different technical choices. However, the authors do not give the formalization for the modeling process. Later, this work was extended in (Dehdouh et al., 2015) by proposing three approaches to implement the DW using a column-oriented NoSQL model. Their approaches are called NLA (Normalized Logical Approach), DLA (Denormalized Logical Approach) and DLA-CF (Denormalized Logical Approach by using Column Family). The first approach (i.e., NLA) uses different tables to store facts and dimensions, and uses a simple attribute for measures and dimension attributes. The second approach (i.e., DLA) proposes to store the fact and dimensions in the same table, and uses a simple attribute to map measures and dimension attributes. The third approach DLA-CF stores the fact and dimensions in the same table, and uses the compound attribute to represent measures and dimension attributes. Nevertheless, the NLA approach is quite inefficient when performing queries with joins. The DLA-CF approach is more efficient than DLA approach only when the query handles attributes belonging to the same dimension.

In (Chevalier et al., 2015a) (Chevalier et al., 2015b) the authors propose a set of rules to map the star multidimensional model into the two NoSQL model: column-oriented and document-oriented. To present the star model into a column-oriented NoSQL model, the authors in (Chevalier et al., 2015c) use the concepts of the column-oriented NoSQL model. Indeed, the star model is transformed into a single table. Since the star model is composed of fact and dimensions, the fact is transformed into a column-family where each measure is presented as a column. And each dimension of the star model is transformed into a column-family where the dimension attributes (parameters and weak attributes) are transformed into a column of the column-family. In the purpose to transform the star model into a NoSQL document-oriented model, the authors of (Chevalier et al., 2015d) define a set of rules in order to transform each star model into a collection of documents. The fact is translated into a composite attribute and each measure is transformed into a simple attribute. As the fact is surrounded by a set of dimensions, each dimension is transformed into a composed attribute (a nested document) and each parameter and weak attribute is converted to a simple attribute. However, the proposed rules do not allow a hierarchical transformation from the star model into the two NoSQL models: column-oriented and document-oriented. In other words, the defined rules do not highlight the hierarchy concept of the star model, which is fundamental for the *DrillDown* and *RollUp* OLAP operations.

Due to the absence of a clear approach that allows the implementation of a NoSQL DW, (Yangui et al., 2016) propose to implement a NoSQL DW. To do so, they propose a set of four rules to transform a star multidimensional model into either column-oriented NoSQL and document-oriented NoSQL models. For each model, they distinguish two types of transformations: *simple* and *hierarchical*.

The first transformation is the mapping to NoSQL model without detailing the hierarchy concept; all hierarchy parameters of a dimension are presented in a single structure into the NoSQL model. It makes a distinction between measures and dimensions; dimensions and facts are stored separately on different column-family/collection. To ensure the links between dimensions and fact, the dimension identifier is duplicated in the column-family/collection representing the fact.

The second transformation (i.e., hierarchical) aims to transform the star model of the data warehouse into a NoSQL model while describing the hierarchy concept. It uses the concepts super-column/document to present dimensions and hierarchies of the multidimensional model (A super-column is composed of a set of columns and a document is defined as a collection of attributes).

Table 1 compares the literature works according to the following criteria. Each criterion *Ci* means that the approach

C1: Transforms the multidimensional model into a column-oriented NoSQL.

C2: Transforms the multidimensional model into a document-oriented NoSQL.

C3: Transforms the star multidimensional model into a NoSQL.

C4: Transforms the galaxy multidimensional model into NoSQL.

C5: Proposes a set of rules to do the transformation into NoSQL.

Table 1: Comparison of works for NoSQL DW.

Approach	C1	C2	C3	C4	C5
(Li, 2010)	-	-	-	-	✓
(Dehdouh et al., 2015)	✓	-	✓	-	✓
(Chevalier et al., 2015a)	✓	✓	✓	-	✓
(Yangui et al., 2016)	✓	✓	✓	-	✓

In this section, we have presented relevant works related to NoSQL DW. We note that some of these works transform the multidimensional model of the DW into a column-oriented NoSQL model whereas some others use the document-oriented NoSQL model. Also, we stress that the proposed approaches were interested in the star model; nevertheless, there is no attempt to transform the galaxy model of document warehouses into NoSQL. This represents a lack we alleviate for the Document Warehouse context using NoSQL.

The remaining of this paper presents our proposed rules to transform a DocW into NoSQL.

3 XML DOCUMENT WAREHOUSE

Documents contain interesting textual data; therefore, they represent a source of elements useful for decisional analyses. Frequently, these documents have XML format and heterogeneous structures even though they share the same domain. The DocW collects and organizes documents to be used by OLAP analyses dedicated for decisional purposes. Due to their heterogeneous structures, the DocW user (*i.e.,* decision-maker) is constrained to write several queries and then manage/merge their results to build the final response. This requires competence and skill to manage this hard task. To alleviate this problem, (Feki et al., 2013), (Pujolle et al., 2011) and (Tournier, 2007) propose approaches to build the schema of the XML DocW. They use the galaxy multidimensional model to describe the schema of the DocW.

The galaxy model can be seen as a network of *entities* (*i.e.,* dimensions) connected by *nodes*. Each node links *compatible* entities. Compatible entities are entities that could be used together in OLAP queries. In a galaxy, each entity can play a double role either an analysis subject (*i.e.,* fact) or an analysis axis (*i.e.,* dimension). The basic concepts of the galaxy model are *nodes*, *dimensions* and *hierarchies*. We formalize these concepts as follows:

Galaxy Model. A Galaxy model is the generalization of the constellation multidimensional model. This model is a grouping of dimensions connected by nodes. The Fact concept is hidden. Formally, a galaxy model *N* is defined by (GM^N, GM^D, GM^N) where:

- GM^N: is the name of the galaxy model.
- $GM^D = \{D_1 ... D_n\}$: is a non-empty set of dimensions.
- $GM^N = \{N_1 ... N_m\}$: is a non-empty set of nodes.

Dimension. A dimension models an analysis axis. It has a set of attributes called parameters organized in hierarchies. Formally, a dimension *D* is a triplet (D^N, D^P, D^H) where:

- D^N: is the name of the dimension *D*.
- D^P: is a set of at least one strong attribute (called parameter) and may have additional weak attributes those label the parameters.
- D^H: is a non-empty set of hierarchies.

Hierarchy. A hierarchy organizes parameters in several levels. It is defined by (H^N, H^P, P^{WA}) where:

- H^N: is the name of the hierarchy.
- $H^P = (P_1, ..., P_q)$: is a set of parameters of H^N.
- P^{WA}: is a function that associates each parameter to its weak attributes.

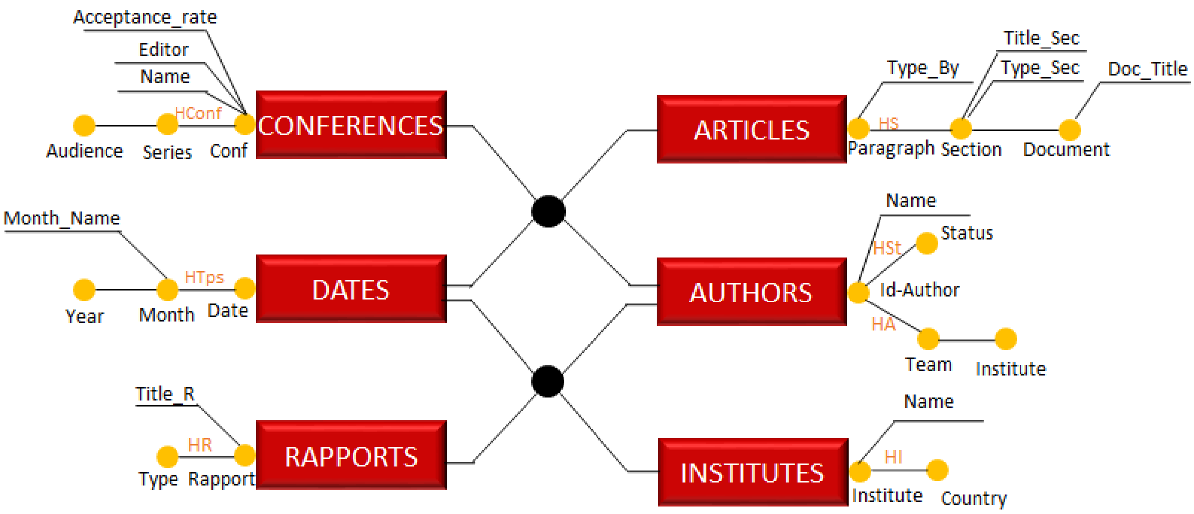

Figure 1: An example of a galaxy model (Tournier, 2007).

Node. A node connects compatible dimensions; *i.e.,* dimension those could be used semantically together within a same multidimensional query. Formally, a node is defined by (N^N, N^D) where:

- N^N: is the name of the node.
- N^D: is a function that links each dimension to the set of its nodes. Naturally, a dimension can be connected to more than one node.

Figure 1 depicts an example of a galaxy multidimensional model, which is composed of six dimensions called *Conferences*, *Articles*, *Authors*, *Dates, Rapports* and *Institutes* connected via two nodes. The first node links the four dimensions *Conferences*, *Dates*, *Articles*, and *Authors*. It describes articles published in a conference and written by authors on a given date. While, the second node connects the four dimensions: *Dates*, *Rapports*, *Authors*, and *Institutes*. It translates reports of projects led by institutes and supervised by scientific personnel (authors) on a specific date. This model allows decision-makers, for instance, to analyze research papers authored by authors and published in conferences.

4 TRANSFORMATION OF A GALAXY MODEL INTO A NOSQL COLUMN-ORIENTED MODEL

In this paper, we assume that a DocW is multidimensional modelled as a galaxy. Because the volume of documents is increasing permanently, looking for new technological solutions that support huge volume of documents is more and more felt both by warehouse users and designers communities. The objective of designers is to better serve decision-makers whose needs are beyond the classical needs of DW users. It is mainly to improving performance of the DocW in terms of response time and queries. In parallel, NoSQL is becoming rapidly an efficient alternate to RDBMSs (Relational Data Base Management Systems) for storing, managing and querying big volumes of data (Chandawni, 2016). This motivated us to explore the usage of this new technology in document warehousing. To do so, we suggest, in this work, dedicated rules for transforming a galaxy multidimensional model of XML documents warehouse into NoSQL. More accurately, we have elected the column-oriented NoSQL model for this work. In fact, this model is the richest NoSQL model and it is characterized by it performance (Lemberger et al., 2015).

To transform a galaxy model into a column-oriented NoSQL model, we define two types of transformations: *simple* and *hierarchical*. The first type of transformation transforms the galaxy model into a column-oriented NoSQL model without detailing the hierarchy concept. Whereas, the hierarchical transformation describes hierarchies when converting the galaxy model into a column-oriented NoSQL model.

In the remaining of this paper, we define the fundamental concepts of the NoSQL column-oriented model and a set of appropriate transformation rules for both simple and hierarchical transformations.

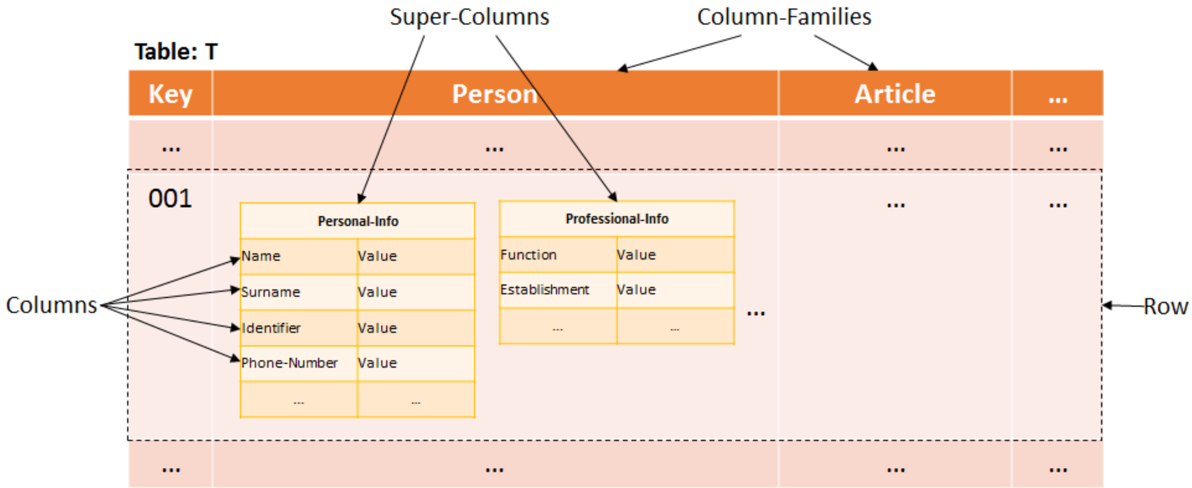

Figure 2: Basic concepts of the column-oriented NoSQL model.

4.1 Column-Oriented NoSQL Model

The column-oriented NoSQL model provides a flexible schema characterized with a large number of columns that can differ between each row without generating null values. For this reason, it can be seen as a set of tables defined row by row. In this section, we define the five basic concepts of this model namely *Table, Row, Column-Family, Super-column and Column*.

Table. A table is a set of lines and a line is composed of a key and a set of column-families. It is defined by (T^N, T^R) where:

- T^N: is the name of the table.
- $T^R = \{R_i, ..., R_n\}$: is a set of rows.

Row. A row represents a record in the table of the database. Formally, it is a couple (R^K, R^{CF}) where:

- R^K: is the key of the row.
- $R^{CF} = \{CF_i, ..., CF_m\}$: is set of column-families.

Column-Family. A column-family consists of a set of columns or super-columns. It is defined by (CF^N, CF^C) where:

- CF^N: is the name of the column-family.
- $CF^C = \{C_1,...,C_p\}$: is a set of columns or super-columns.

Super-Column. A super-column allows the grouping of columns with data semantically linked (Lemberger et al., 2015). It is defined by the couple (SC^N, SC^C) where:

- SC^N: is the name of the super-column.
- $SC^C = \{C_1,...,C_q\}$: is a set of columns.

Column. A column is characterised by a name and an atomic value. We note that each atomic value can be historised with a time label: timestamp. In fact, this principle is useful for historical management (Wrembel, 2009). Formally, a column is a couple (C^N, C^V) where:

- C^N: is the name of the column.
- C^V: is the column value.

Figure 2 shows the basic concepts of the column-oriented NoSQL model.

4.2 Simple Transformation

The simple transformation aims to convert the concepts of the galaxy model into a NoSQL model but without detailing the hierarchy concept; transforming parameters of a dimension uses a unique structure of the NoSQL model. To achieve this goal, we define a set of four rules.

Rule 1: Each galaxy model G transforms into a table T in the NoSQL column-oriented database.

Obviously, the galaxy is the most generic concept of a multidimensional galaxy model; consequently, it can represent only a table that presents the global concept in the column-oriented NoSQL model.

Rule 2: Each node N belonging to a galaxy G and connected to a set of compatible dimensions $\{D_1, ..., D_n\}$ transforms into a column-family CF.

Remember that in a NoSQL column-oriented model, the column-family is a concept that contains a set of columns. In addition to this, in the galaxy model, a node is connected to a set of dimensions. Consequently, the node merits to be transformed into a column-family and each dimension connected to the node is presented by a column.

Rule 3: Each dimension D_i of the galaxy is transformed into a column-family CF_{Di} where each attribute (parameters and weak attributes) of D_i is transformed into a column C_i of CF_{Di} ($C_i \in CF_{Di}$).

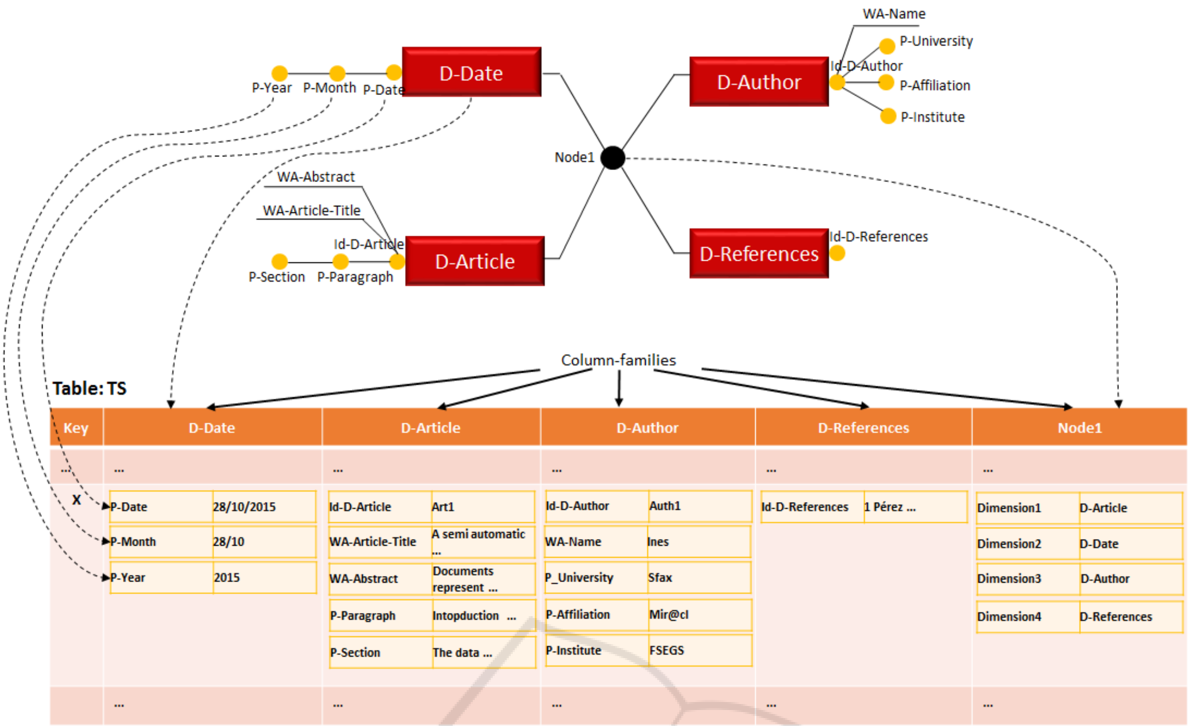

Figure 3: Example of a row in a table (Simple transformation).

In a galaxy, a dimension is characterized by a set of attributes. In the NoSQL column-oriented model, the column-family contains columns. Thereafter, each dimension of the galaxy is transformed into a column-family and each dimension attribute is transformed into a column.

Rule 4: The nodes and dimensions instances are transformed into a row of the table of the column-oriented NoSQL model.

Recall that a row in a table of the column-oriented NoSQL model is composed of a key, a set of column families denoted *CF*. Consequently, the instances of node, and the associated instances of dimensions transform into a row of the table of the column-oriented NoSQL model.

Figure 3 depicts an example of a row in a table of the NoSQL column-oriented database. This row is obtained by applying the simple transformation rules on a galaxy composed of four dimensions: *D-Date, D-Author, D-Article* and *D-References*, and a single node. We note that this transformation does not take into account hierarchies dimensions. To do so, we propose a hierarchical transformation in order to describe parameters hierarchies in the column-oriented NoSQL model.

4.3 Hierarchical Transformation

Unlike the simple transformation, the hierarchical transformation explain the hierarchy concept when transforming the multidimensional model into a column-oriented NoSQL model. This transformation exhibits parameters of each hierarchy into a separate structure of the NoSQL model. For the hierarchical transformation, we keep three rules from the simple transformation (Rules 1, 2 and 4) and one specific rule we have defined for this transformation.

Rule 5: Each dimension D_i of the galaxy transforms into a column-family CF_{Di} where each hierarchy is transformed into a super-column. The name of the super-column is the name of the hierarchy, whereas parameters and weak attributes become columns.

In a column-oriented NoSQL model, a column-family is composed of a set of super-columns and a super-column is composed of columns. In the other hand, in a galaxy model, a dimension is composed of a set of hierarchies, and a hierarchy is composed of a set parameters. Thereafter, we can use the concepts column-family, super-column and column to present dimensions and hierarchies into the column-oriented NoSQL model.

Figure 4 shows an example of a row in a table of the NoSQL column-oriented database obtained by applying the hierarchical transformation rules on the galaxy at the top of the figure.

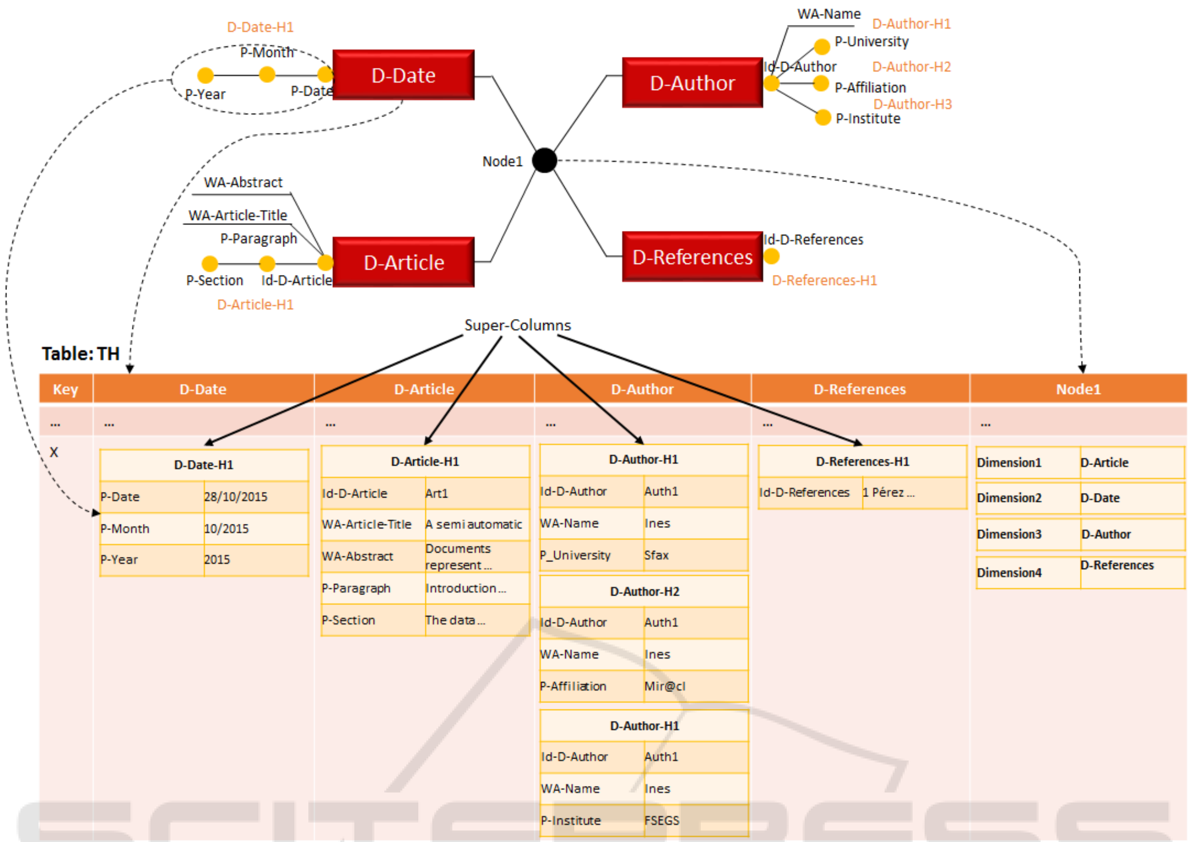

Figure 4: Example of a row in a table (Hierarchical transformation).

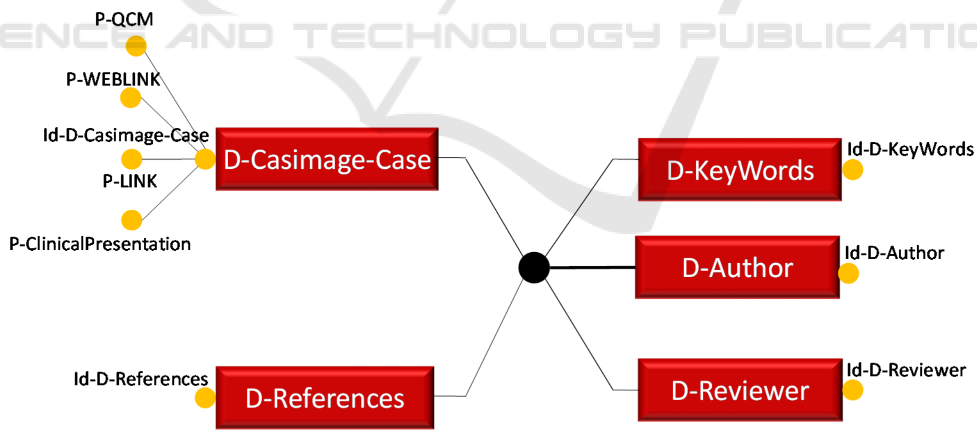

Figure 5: Galaxy model for the medical collection (Ben Messaoud et al., 2015).

5 EXPERIMENT AND EVALUATION

Due to the absence of a benchmark for the galaxy model, we conduct our experiment on the galaxy model generated from a set of 1691 XML documents issued from the medical collection Clef-2007 and described by three DTDs (Ben Messaoud et al., 2015). This galaxy model is composed of five dimensions: *D-Casimage-Case*, *D-Author*, *D-Keywords*, *D-Reviewer* and *D-References*. Figure 5 presents galaxy model for the medical collection. We note that weak attributes are voluntary withdrawn to simplify the galaxy model.

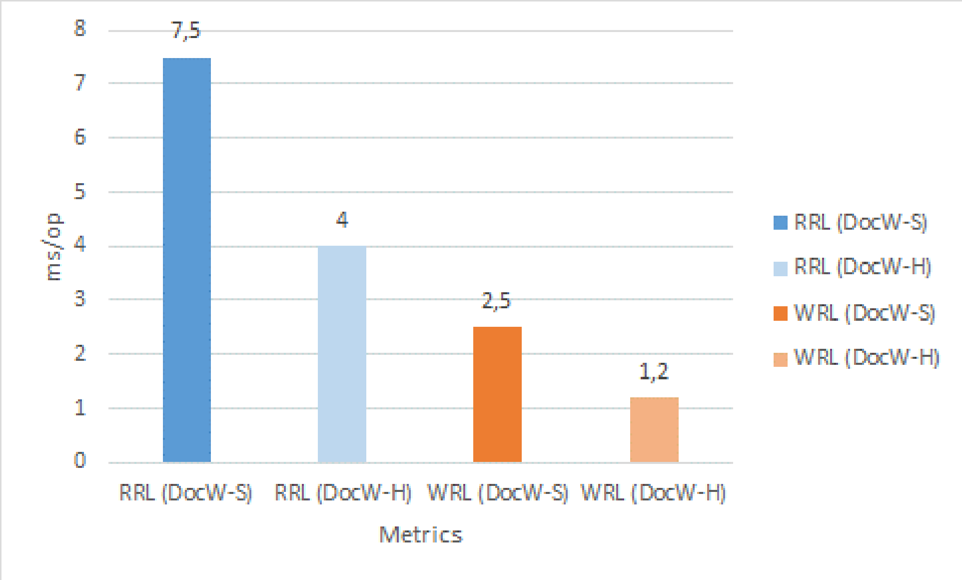

Figure 6: Evaluation of the implemented NoSQL column-oriented XML DocWs.

In order to implement the galaxy XML Document Warehouse as a column-oriented NoSQL model, we have elected Cassandra as a NoSQL database management system. In fact, it can manipulate a very large amount of data like any NoSQL database management system. It includes the concepts of columns, super-column and column-families that we have used in our proposed rules.

To define the schema of the NoSQL DocW with Cassandra, we used the Cassandra Query Language (CQL). By using this language, it is useful to create table, column-family, etc.

In the context of our work, we distinguish two NoSQL DocW: *DocW-S* and *DocW-H* implemented by applying respectively the simple and hierarchical transformation rules.

To load the data of the galaxy model into the NoSQL column-oriented DocW, we opted for the use of the Talend Data Integration tool. This tool allows extracting data from large, heterogeneous data sources and integrating them into the NoSQL database. In the context of our work, data integration is carried out in accordance with our transformation rules.

In order to evaluate the two-implemented NoSQL document warehouses, we use the two metrics: *Write Request Latency* (WRL) and *Read Request Latency* (RRL). The WRL metric assesses the speed of the system during the data loading stage. The RRL metric measures the response time of the system to answer user requests.

Figure 6 illustrates the evaluation of the two implemented NoSQL XML DocW.

By comparing the values obtained with each metric on the two implementations (Simple and Hierarchical) of the DocW, we conclude that the DocW implemented using the hierarchical transformation is better than the simple one. Indeed, the loading time and the user request response time for the DocW-H are better than the DocW-S; To load data in DocW-S, we need approximately the double of loading time of DocW-H and to answer to decision maker queries with the DocW-S we need more than the twice of the response time of DocW-H.

6 CONCLUSION AND FUTURE WORKS

Documents represent an important source of information for decisional analyses. So, they merit to be integrated in the decision support system. Likewise, documents should be warehoused. As a result, the document warehouse concept has emerged. Nevertheless, the increasing production of documents, the sharing of information between users and the diffusion of data via networks generate a very large volumes of available and interesting data to be analysed. This huge amount of data require appropriate storage means (Favre et al., 2013).

The NoSQL environment provides a solution to answer the limitations of the relational systems in terms of scalability and handling a large volume of data. To benefit from this new technology, we proposed, in this paper, an approach to build a NoSQL Document Warehouse. More accurately, we transform the multidimensional galaxy model of the DocW into the column-oriented NoSQL model. From NoSQL models, we elected the column-oriented model because its performance has been proven in the literature works.

To build a NoSQL DocW, we distinguish two transformation types: *simple* and *hierarchical*. The first transformation converts the concepts of the galaxy model into a NoSQL model without detailing the hierarchy concept. For this transformation, we define a set of four rules. While, the hierarchical transformation explains the hierarchy concept when transforming the multidimensional model into a column-oriented NoSQL model. It retains three rules from the simple transformation and defines one specific rule for this transformation.

To substantiate these rules, we use the NoSQL database management system *Cassandra* and *Cassandra Query Language* (CQL) to apply the simple and hierarchical transformation rules. We obtain respectively DocW-S and DocW-H. Moreover, the evaluation of the obtained NoSQL DocW in terms of the two metrics *Write Request Latency* and *Read Request Latency* on a medical collection shows that the DocW-H is better than the DocW-S.

As a future work, we will propose rules to transform the multidimensional model of the document warehouse into the document-oriented NoSQL model and compare the performance of the two NoSQL DocWs: column-oriented DocW and document-oriented DocW. In addition, we expect define a set of analytical operations dedicated to the galaxy model of the NoSQL DocW.

REFERENCES

Agrawal, D., Das, S., El Abbadi, A., 2011. Big data and cloud computing: current state and future opportunities. In *EDBT/ICDT'11, 14th International Conference on Extending Database Technology,* pp. 530–533.

Ben Mefteh S., Khrouf K., Feki J., Ben Kraiem M., Soule-Dupuy C., 2016. A Semantic Approach for XML Document Warehousing and OLAP Analysis. In *IJIDS'16, International Journal of Information & Decision Sciences*, vol.8, n°.3, pp. 254-283, DOI: 10.1504/IJIDS.2016.078587.

Ben Messaoud, I., Feki, J., Zurfluh, G., 2015. A Semi-automatic Approach to Build XML Document Warehouse, in CCIS'15, *Communications in Computer and Information Science*, Springer International Publishing Switzerland 2015, A. Fred et al. (Eds.), pp. 347–363.

Chandawni, G., 2016. NOSQL DATA-WAREHOUSE. In *IJIRCCE'16, International Journal of Innovative Research in Computer and Communication Engineering,* Vol. 4, Special Issue 4, pp. 96-104.

Chevalier, M., Elmalki, M., Kupliku, A., Teste, O., Tournier, R., 2015a. Entrepôts de données multidimensionnelles NoSQL. In *EDA'15, 11ème Journées francophones sur les Entrepôts de Données et l'Analyse en ligne*, p.161-176

Chevalier, M., El Malki, M., Kopliku, A., Teste, O., Tournier, R., 2015b. Implementing multidimensional data warehouses into NoSQL. In *ICEIS'15, 17th International Conference on Enterprise Information Systems*, pp.108-130.

Chevalier, M., El Malki, M., Kopliku, A., Teste, O., Tournier, R., 2015c. Implementation of Multidimensional Databases in Column-Oriented NoSQL Systems. In *ADMIS'15, 19th East-European Conference on Advances in Databases and Information Systems,* pp. 79-91.

Chevalier, M., El Malki, Teste, O., Tournier, R., 2015d. Implementation of Multidimensional Databases with Document-Oriented NoSQL. In *DAWAK'15, 17th International Conference on Big Data Analytics and Knowledge Discovery,* pp. 379-390.

Dehdouh, K., Bentayeb, F., Boussaid, O., 2014. Columnar NoSQL Star Schema Benchmark. In *MEDI'14*, *4th Model and Data Engineering,* pp. 281-288.

Dehdouh, K., Bentayeb, F., Boussaid, O., Kabachi, N., 2015. Using the column oriented NoSQL model for implementing big data warehouses. In *PDPTA'15*, *21st International Conference on Parallel and Distributed Processing Techniques and Applications*, pp.469-475.

Favre, C., Bentayeb, F., Boussaid, O., Darmont, J., Gavin, G., Harbi, N., Kabachi, N., Loudcher, S., 2013. Les entrepôts de données pour les nuls ou pas !. In *EGC'13, 13th conference francophone sur l'extraction et la gestion de connaissance,* pp. 1-18.

Feki, J., Ben Messaoud, I., Zurfluh, G., 2013. Building an XML Document Warehouse. In *JDS, Journal of Decision Systems*, Vol. 22 No. 2. pp. 122-148, DOI: 10.1080/12460125.2013.780322.

Jacobs, A., 2009. The pathologies of big data. In *Communications of the ACM 52(8)*, pp. 36–44.

Lemberger, P., Batty, M., Morel, M., Rafaelli, JL, 2015. *Big Data et Machine Learning*, Dunod. 1st edition, ISBN: 978-2-10-072074-3.

Li, C., 2010. Transforming relational database into HBase: A case study. In *ICSESS'10, International Conference on Software Engineering and Service Sciences*, pp. 683–687.

McCabe, C., Lee, J., Chowdhury, A., Grossman, D., Frieder, O., 2000. On the design and evaluation of a multi-dimensional approach to information retrieval. In *SIGIR'00, 23rd International Conference on Research and Development in Information Retrieval*, pp. 363-365.

Pujolle, G., Ravat, F., Teste, O., Tournier, R., Zurfluh, G., 2011. Multidimensional database design from document-centric XML documents. In *DAWAK'11, 13th International Conference on Data Warehousing and Knowledge Discovery*, pp. 51–65.

Stonebraker, M., 2012. New opportunities for new sql. Commun. In *ACM 55(11),* pp. 10–11.

Tournier, R., 2007. Analyse en ligne (OLAP) des documents. Thèse de doctorat en Informatique, Université Toulouse III, Paul Sabatier, Toulouse, France.

Tseng, F. S. C., Chou A. Y. H., 2006. The concept of document warehousing for multi-dimensional modeling of textual-based business intelligence. In *Decision Support Systems*, pp. 727-744.

Wrembel, R., 2009. A survey of managing the evolution of data warehouses. In IJDWM'09, International Journal of Data Warehousing and Mining, pp. 24–56.

Yangui, R., Nabli, A., Gargouri, F., 2016. Automatic Transformation of Data Warehouse Schema to NoSQL Data Base: Comparative Study. In *KES'16, 20th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems,* pp. 255-264.