International Journal of Computer Applications (0975 - 8887)
Volume 175 - No.36, December 2020

Multidimensional and Non-Relational Data Models: A Comparison with a Big Volume of Data

Maria Camila S. de Lira
Federal Rural Univesity of Pernambuco
Dom Manuel Medeiros Street, Dois Irmaos
Recife-PE/Brazil

Ademir B. Santos Neto
Federal Rural Univesity of Pernambuco
Dom Manuel Medeiros Street, Dois Irmaos
Recife-PE/Brazil

Maria C. Moraes Batista
Federal Rural Univesity of Pernambuco
Dom Manuel Medeiros Street, Dois Irmaos
Recife-PE/Brazil

Roberta Macedo M. Gouveia
Federal Rural Univesity of Pernambuco
Dom Manuel Medeiros Street, Dois Irmaos
Recife-PE/Brazil

Tiago Alessandro E. Ferreira
Federal Rural Univesity of Pernambuco
Dom Manuel Medeiros Street, Dois Irmaos
Recife-PE/Brazil

ABSTRACT

In the last decade, a huge volume of data has been created every day. There are many sources where this information comes from, among them, it is possible to cite: financial transnational records, internet banking, call centers, ATMs, sensors, among others. As the amount of data is increasingly really quickly and the importance of the information inside the big data is crucial to the world, new tools and technologies are appearing to support the use and the manipulation of big amounts of data. This paper presents a comparative analysis of two data models, a multidimensional data model (Data Warehouse) and a non-relational data model, also known as NoSQL, in order to show their efficiency about insert and query the data in a context of big volumes of data

Keywords

Comparison, Data Warehouse, HBase, Hadoop, NoSQL

1. INTRODUCTION

Information is a hidden treasure in an amount of data. In almost all cores of the real world, it is necessary a huge amount of data to obtaining relevant information. Not long ago, according to Alliance [1], data collecting requires patterns for hundreds of years to discover any rain pattern. It means to take a sit on the road and take notes about the traffic's speed to plan the transport network. Involved in collecting thousands of medical handwritten notes to check how the diseases work and how to heal them. However, today the data is being generated by sensors present in millions of devices, vehicles, machines, and even in street poles. There are many sources of information, such as messages, updates, and images posted to social networks; readings from sensors; GPS signals from cell phones; financial transactions records, and more [2].

According to Gantz and Reinsel [3] the estimated volume of data generated in the world was 1.8 zettabytes what means about 10^{21} bytes, and this number tends to double every two years. According to Wang et al. [4] the world generates 2.5 Quintilian of data daily. Making a simple comparison, throughout 1992, the global internet traffic was 100 Gigabytes per day [5]. All this volume of data, coming from a variety of sources and in high velocity is a phenomenon called Big Data.

The management of great volumes of data is transforming many areas of society, for example, healthcare, science, engineering, finance, business, among others because of the new possibilities that its analysis is creating [6]. The information inside this big amount of data is really important and many tools have been developed to analyze this data. According to Sagiroglu [7] the process of investigating a big quantity of data to reveal hidden patterns and secret correlations is called "big data analytics". This information is really useful because promotes rich and deep insights which leads to important knowledge.

The internet is a huge repository with many unstructured textual information [8]. The traditional relational model is no longer proper to work with big data because it does not have enough flexibility to store huge volumes of data and manipulate unstructured data [9]. Han et al. [10] show some limitations of this model such as slow writing and reading with the risk of dead-locks and parallel problems, limited capacity to work with social network services, and the technical problems about expanding the number of relationships in databases with many tables. Therefore, there are important solutions that have high flexibility, performance, support any kind of data and good storage [11]. Considering these characteristics, the use of different data models from the relational model has been studied in order to get tools that support issues about work with big volumes of data. There are some technologies to manipulate and

analyze a big quantity of data, among them, there are the multidimensional model and the non-relational model.
Multidimensional modeling is the technique to model the database to aid queries in a Data Warehouse (DW), which is a database projected to queries and analysis of a big volume of data instead of the transaction processing (Lane and Schupmann [12]). The non-relational model can be more adequate to solve problems as treatment of big volumes of data, executions of queries with low latency, and flexible models, as documents in Extensible Markup Language (XML) or JavaScript Object Notation (JSON) format. A trend to solve many problems and challenges generated by the big data context is the NoSQL (not only SQL) movement (Vieira et al. [13]).
The main aim of this paper is to make a comparative analysis between the multidimensional model (Data Warehouse) and the non relational model. This article aim to discover which solution works better in the context of big volume of data obtained from sales operations. There were analyzed some metrics, such as: storage, flexibility and performance. In this paper was utilized a huge volume of XML documents with data from some market sales operation.

2. DATA MODELS

According to Liu et al [14] a Database Management System (DBMS) is a collection of data not related and a set of software to access these data. The collection of data has relevant information for the company. The main aim of the DBMS is to offer a solution to store and get information from the database in a convenient and efficient way.
A data model has a description of the data hiding many low-level details of the storage. It means that the model describes the types of information that are stored in the database. In summary, a data model is a collection of conceptual tools to describe the data. There are several data models [15], among them are:

—The relational model for database management is an approach to managing data using a consistent structure and language with first-order predicate logic where all data is represented in terms of tuples, grouped into relations.

—The multidimensional model is an integral part of the On-Line Analytical Processing (OLAP). It is projected to solve complex queries in real-time. The relational implementation of the multidimensional model is usually the star scheme, whereby conversion organizes the data in the dimensions and fact tables. This data model will be better detailed in a further Section.

—The non-relational model is a database that does not incorporate the traditional relational model. This kind of model is scheme free, promoting high availability and scalability. This data model will be better detailed in a further Section.

2.1 Multidimensional Model

The multidimensional model clarifies the understanding and visualization of typical problems in decision support, and it is more intuitive to analytically process. According to Pedersen [16] the focus in the multidimensional model is a collection of numeric business measures. Each measure depends on a set of dimensions. Each row in the dimension tables can describe many registers in the fact table. Kimball et all [17] says that a multidimensional model is made by a central table (fact table) and a set of other tables interconnected (dimension tables). The fact table is metaphorically seen as a cube because all the dimensions coexistent for all points in the cube and they are all independents. A fact table has measurements of the business or records events [18].

The Data Warehouse (DW) is a tool used to store a huge quantity of data, usually from transitional systems. The main aim of the DW is helping the user to query the data stored in the DW to take decisions. The content in the DW must be flexible, consistent, and safe. Data warehouse organizes and stores the data needed for informational, analytical processing over a long historical time perspective [19]. According to Immon [20] a DW is a collection of oriented data to topics, integrated, non-volatile, and varying in time. There is a tool for queries in multidimensional databases with a set of applications called OLAP (on-Line Analytical Process) which is characterized by dynamic multi-dimensional analysis of consolidated enterprise data supporting user analytical [21]. The OLAP and the DW work together because of the DW stores the information in an efficient way the OLAP should get them efficiently. Some OLAP tools have the capacity to manipulate and analyses big volumes of data under many perspectives. These tools enable managers to have different visions of the same data set [22].

2.2 DW and OLAP

Data warehousing is the integration process to corporative data from some company in a single repository. This is a supportive environment to support decisions that handle data stored in several sources, organize them, and give the analysis to the decision-makers. This is the technology to manage and execution of analysis over the data.
The OLAP tool enables analysts, managers, and executives to have quick, consistent, and interactive access to a huge variety of possible views of the information. With an OLAP tool, there are possible answer questions such as: who are the 10 worse sellers from the branch X, what is the average salary of the its employees in the company, what is the volume of sales of the product Y in branch Z. Beyond these characteristics before cited, there are differences in the type of application that is used for each tool. Transactional system applications, web services, and client-server use OLTP (Online Transaction Processing). Managers, executives, and data scientists usually use OLAP. According to DataOnFocus [23], OLAP and OLTP are distinct in the use of the database. OLAP is more focused on the analytical process and generates complex graphs and reports. OLTP focus on an optimized transactional system, supporting a huge quantity of changes in the data and more simple reports.

2.3 Non relational Model

According to Han et al. [10] the continuous development of cloud computing and the internet, bring a need for databases that can support: high concurrence in reading and write with low latency, efficient storage to big volumes of data, high scalability, and availability and low operational cost. In agreement with Brito [24] because of these needs began to emerge databases that can supply this demand. Thinking about this, the projectionist of databases start to develop storage strategies that can be free from some rules from the relational model. Brito [24] says that the term NoSQL (not only SQL) was used to represent solutions where the relational model was not more adequate. According to Cattell [25] the main characteristics of the NoSQL database is the high performance of parallel storage on a big scale.
MapReduce is the main paradigm used in NoSQL databases. According to Vieira [13] it was initially proposed to simplify the processing of big volumes of data in parallel and distributed architecture, as clusters, for example. The main aim of MapReduce is to make transparent the distribution details of the data and balance the charge, allowing the programmer to focus only on the treatment of the data. According to Corporation [26], MapReduce is a program-

ming paradigm that allows the processing of big volumes of data. The MapReduce is the union of two different tasks the map and the reduce. The *map* convert the set of data into another set where the individual's elements are divided into tuples (Key/Value). The *reduce* has as input and output the combination of tuples that come from the map in a smaller set of tuples.

According to Vieira *et al.* [13], due to the consistency of the model in the traditional DBMS be strongly related to transactional control ACID, a huge increase of data is unfeasible. As NoSQL databases and their process are distributed in several nodes, this transactional control is practically impractical. The CAP (Consistency, Availability and Partition tolerance) theorem ([27]), says that there are three properties that are important to the NoSQL database: Consistency means that all users can see the update operations that was made by someone in the database, availability of the system even in failures situations. Partition is related to the system's continuing to work even if a node presents a failure. The CAP theorem says that is not possible to get all the tree objectives simultaneously, and hence one of the properties must be despised. Following this theorem, the NoSQL databases use the BASE (Basically Available, Soft-state, Eventual Consistency) paradigm to control the consistency. According to Strauch et al. [28], the BASE proprieties work as an application that works full time (basically available), do not must be consistent all the time (soft-state), but will eventually in a known state (eventual consistency). This property allows that the date is distributed in different repositories.

There are four main categories of NoSQL databases according to Moniruzzaman and Hossain [29]:

—Key-Value - This model represents the database as a hash table and it is composed of unique keys. This model has easy implementation and so allows the data to be accessed quickly by its key, especially in a system that has high scalability. This approach is recommended when it is necessary to access many data where this data can vary from register to register. This model is receiving increased attention from the research community in an effort to improve their performance and scalability [30].

—Columns - In this model, the data are indexed in a triple (line, column, and timestamp), where lines and columns are identified by keys and timestamp allows differentiate multiple versions of the same data. In this category, there is the column family, which is used to group the columns which store the same type of data. In this model, the main characteristics are persistence, data partitioning, and strong consistency. Reading a subset of a tables columns becomes faster, at the potential expense of excessive disk-head seeking from column to column for scattered reads or updates [31].

—Documents - According to Brito [24], in this model the documents are basic units of storage and do not necessarily use any type of predefined structure, as tables in the relational model. A document is an object which has a unique identifier that consists of certain fields. This document is in JSON (Javascript Object Notation) format.

—Graphs - This model has basically three components: the nodes (vertices in the graph), the relationships (the edges), and the properties (attributes) of the nodes and relationships according to Nayak et al [32]. Each node can be connected by more than one edge. The information is classified and stored as entities, as the relationships are established by connections. This model is flexible and can be expanded through many machines. This model has the advantage to perform complex queries. This kind of model is recommended for real-time applications due to its good performance in queries, mainly with social network data.

There are many NoSQL Databases. As an example it is possible to cite: MongoDB [33], Dynamo [34], Voldemort [35], Cassandra [36], Memcached [37], HBase [38] etc. The software library Apache Hadoop is a framework that allows the distributed processing of a big volume of data through cluster computing. The Hadoop project includes 4 modules, they are: Hadoop Common, Hadoop Distributed File System (HDFS) [39], Hadoop Yarn, and Hadoop MapReduce. The HBase is a distributed and scalable database to store big volumes of data. This is a database-oriented to columns where the columns are grouped in column families and stored together on the disk. Also, it is used to aleatory access to read and write in the application's data. The goal of this project is to store very big tables - billions of lines for million of columns - under a cluster. The HBase offer distributed storage of the data for the Hadoop. The Hive [40] is the Hadoop Data Warehouse structure that provides data summarization and ad-hoc queries. It eases the read, writes, and manipulation of big sets of data, in a distributed architecture using SQL, allowing tasks for example Extraction, Transformation and Load (ETL), reports, and data analysis.

3. METHODOLOGY

In this Section is described the development process of the two models which will be compared.

3.1 Data Selection

This paper used data obtained from sales operations, more specifically, data from purchase and sale operations in the metropolitan region of a city in the northeast region of Brazil. These operation data initially are in XML format. The Figure 1 shows part of one of this operation data in XML format. A total of about 3.000.000 operation data in XML format were used in this study. In the first attempt to manipulate the data, the XML files were transformed into structured data, to a relational DBMS. The machine used for this task was a standard computer (with a 2.2GHz Intel Core i5 CPU, 16 GB RAM, and 750 GB hard disk).

<?xml version="1.0" encoding="UTF-8"?><?xml-stylesheet type="text/xsl" href="xsl/an/_Visualizacao_Resumo.xsl"?><nfeProc xmlns="http://www.portalfiscal.inf.br/nfe"><mostraIP>0</mostraIP><numIP>null</numIP><NFe xmlns="http://www.portalfiscal.inf.br/nfe"><infNFe Id="NFe26151000118694000166550010000049571000049572" versao="3.10"><ide><cUF>26</cUF><cNF>00004957</cNF><natp>Venda a vista</natOp><indPag>0</indPag><mod>55</mod><serie>1</serie><nNF>4957</nNF><dhEmi>2015-10-26T09:08:00-03:00</dhEmi><dhSaiEnt>2015-10-26T09:08:00-03:00</dhSaiEnt><tpNF>1</tpNF><idDest>1</idDest><cMunFG>2611606</cMunFG><tpImp>1</tpImp><tpEmis>1</tpEmis><cDV>2</cDV><tpAmb>1</tpAmb><finNFe>1</finNFe><indFinal>1</indFinal><indPres>1</indPres><procEmi></procEmi><verProc>f-2015.0.1.6</verProc></ide>

Fig. 1: Streetch of the invoice in XML format

In order to make this manipulation in the data, was used a script written in Python language [41]. The data information about the sellers, costumers, address of the seller, information related to transportation, volume of product, taxes, market segment, product segment and others. In the end of the transformation process were created in MySQL the following tables: *cnae*, *destinatario*, *emitente*, *emitente_distancia_destinatario*, *imposto*, *imposto_item_nf*,

Table 1. : Records and attributes to each table in the relational database

Table	Records	Attributes
cnae	1.329	11
destinatario	139.468	23
emitente	120	24
emitente_distancia_destinatario	170.279	5
imposto	4.415.338	13
imposto_item_nf	14.509.220	4
item_nf	2.901.844	28
municipio	5.570	5
ncm	12.498	5
nfe	686.580	33
produto	171.105	7
similaridade	24.568	8
tipo_imposto	5	2
tipo_produto	3132	2
totais	686.651	17
transporte	2681	11
volume	686.651	8

municipio, *ncm*, *nfe*, *item_nf,produto*, *similaridade*, *tipo_imposto*, *tipo_produto*, *totais*, *transporte* and *volume*. Some of this table have a big number of columns like the table *nfe* that has 33 columns. Other have a large number of register as the table *imposto_item_nf* with more than 14 millions of registers. The Table 1 shows the quantity of records (lines) and attributes (columns) in each table of the relational database.

3.2 Generation and Load in Data Warehouse

After create the relational model starts the phase of ETL (extraction, transformation and load) in the DW. For this was created the dimensions tables and the fact table according to the star structure model [42]. The following dimensions were created: *dim_data* (with data about time), *dim_destinatario* (with data about the costumers), *dim_emitente* (with data about the sellers), *dim_produto* (with data about the products), *dim_segmento_empresa* (with data about the company segment), *dim_segmento_produto* (with the data about the product segment), *dim_dest_localidade* (with data about the locality of the costumer) and *dim_emit_localidade* (with data about the locality of the seller). The fact table created was *fato_volumoe_vendas* to see the volume of sells.
In order to insert the data in the DW tables were used the software Pentaho Data Integration (PDI)[1]. This software is responsible to migrate data between applications, export data from databases, and cleaning. The process of load in the Pentaho is through the creation and transformation of data. These transformations are archives where are defined records inserted in the dimension table. A transformation is a set of interconnected steps that contain the sources of input and output of the data. Table 2 shows the times to the creation and the total numbers of registers to each dimension.
After load, the 8 dimensions start the process to load the table fact. The steps evolved in the table fact are: table input (select the information in the operational base), database lockup (create the relations between the primary keys and the keys of the dimension), remove values (remove the primary keys of the operational base, memory group by (group values of the dimensions keys) and table output (insert the data in the table fact). However, because the computational resources are available, it was not possible to make

[1]Pentaho is available in http://www.pentaho.com/

Table 2. : Table with the consumed time and total of registers for each dimension

Dimension	Execution time	Registers
dim_segmento_empresa	4.3s	1.330
dim_segmento_produto	1min 9s	12.499
dim_produto	2h 22min	171.106
dim_emitente	0.3s	121
dim_emitente_localidade	0.1s	121
dim_destinatario	2h 50min 42s	139.469
dim_dest_localidade	1min 1s	139.149
Total	**5h 15min 6s**	**464.115**

the load of the fact using the PDI. It was noticed that when the number of rows gets 100.000 lines the PDI runs out of memory and crashed. The first idea was to allocate more memory to the Pentaho. However, after 111 hours of execution the machine reboot by itself. Table 3 shows details about PDI just before the system reboot.
The other two attempts were tried. The same scheme in a machine with more resources, but it did not work as well. Another attempt was to parallel the steps of the transformation. However, after 120 hours of execution the machine reboot by itself once again. Therefore the solution that worked did not use the PDI to load the table fact because the computational resources were not compatible with the problem. The step *table output* was created manually in the operational base and the result imported from a *.csv* file. The time to execute the selection was 11 minutes and 59 seconds. The tool used to do this process was the SQLyog.
This file has 2.901.844 lines. The next step was to load the table fact. The load of the *.csv* in the fact table took 38min 1s. The total time of the creation of the DW was 6 hours, 5 minutes, and 6 seconds.

3.3 Generation and Load in NoSql Database

The software used was HBase, which is the database from the framework Hadoop. The first step is to install the Hadoop. The generation and load were made in the Xubuntu system running in a virtual machine through the software VMWare Workstation. After the Hadoop was installed the database HBase was installed as well. Sqoop is a tool in the Hadoop used to import data between structured database and Hadoop. The import process using the Sqoop tool worked well for a while, but when the table with many records would be imported, the tool could not do the task. The Sqoop was used to import the data from these tables: *cnae*, *destinatario*, *emitente*, *municipio*, *emitente_distancia_destinatario* and *ncm*. In order to import the others, tables were used the ImportTsv which is a utility of the HBase to load data in TSV[2] in the database. To use this tool was the need that the data was in a file with TSV format, to be inserted in the HDFS, and finally, be inserted in the HBase. The steps involved in the import of the table in the relational database to the Hbase are export the data from the relational model to a TSV file through a SQL, insert the TSV file in the HDFS, and finally, execute the command to import the file that is in the HDFS in the HBase. Table 4 shows the consumed time of each import of table in the HBase.

[2]This is a format where each register is a line and each register in this line is separated by a tab space

Table 3. : Second attempt to load the table fact

Step Name	Read	Written	Input	Status	Time	Speed (r/s)
General Select	0	2901844	2901844	Finished	14h 51m 58s	54
dim_emitente	2901844	2901844	2901844	Finished	14h 51m 58s	54
dim_destinatario	2386894	2386893	2386893	Running	**110h 55m 57s**	6
dim_segmento	2386893	2386893	2386893	Running	**110h 55m 57s**	6
dim_produto	2386894	2386893	2386893	Running	**110h 55m 57s**	6
dim_des_localidae	1174851	1174851	1174851	Running	**110h 55m 57s**	3
dim_emit_localidae	1174851	1174851	1174851	Running	**110h 55m 57s**	3
dim_tempo	1174851	1174851	1174851	Running	**110h 55m 57s**	3
Remove values	1174851	1174851	1174851	Running	**110h 55m 57s**	3
Memory group by	1174851	0	0	Running	**110h 55m 57s**	3
Table outputs	0	0	0	Running	**110h 55m 57s**	0

Table 4. : Consumed time of each import of tables in the HBase

Table	Import time
imposto_item_nf	1h 17min 27s
imposto	37min 55s
emitente_distancia_destinatario	44s
emitente	36s
destinatario	1min 34s
cnae	55s
ncm	1min 5s
item_nf	2h 24min
nfe	1h 16min 12s
municipio	1min 1s
tipo_produto	6s
produto	24s
Total	**5h 41min e 57s**

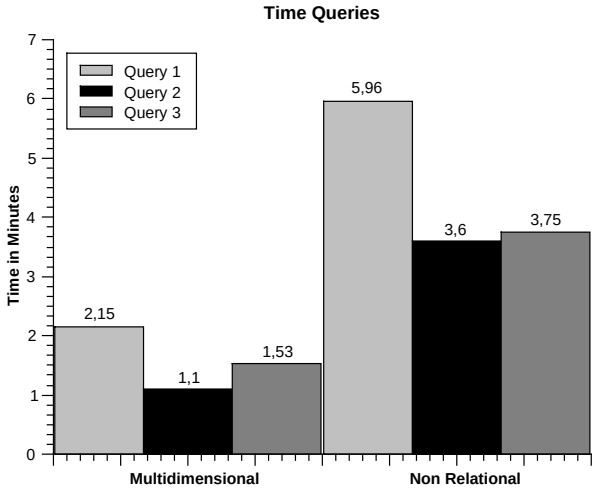

Fig. 2: Comparison between the consumed time in the queries for the two studied models

4. RESULTS

In this paper, the multidimensional model and the non-relational model are compared about their efficiency in two different features, performance in queries and loading. The HBase is a non-relational database-oriented to columns. On this, only one table by time can be read in this database. However, the Hive promotes a Data Warehouse structure that can be used to execute queries in the HBase through the Hive HBase Integration.

To make the consults in the multidimensional model was used the OLAP Saiku.

The first query in the two models was the total amount of sales and the displacement time of the customer, grouped by companies, month, and year. The second query was the total amount of sales, units sold and the displacement of the customer separated by year. The third query was the total amount of units sold, grouped by the companies segment which sold the product separated by the year. The data in the queries are the same in the two models. However, the consumed time in the multidimensional model was smaller than in the Hbase. Also, the number of lines was much smaller in the DW than in the non-relational model. This happens because the OLAP groups the information on many levels in such a way that the number of lines returned reduce, optimizing the query result. Table 4 shows the time consumed and the total number of lines for each one of the tree queries previously described. Figure 2 allows a more intuitive comparison of the information presented in Table 5. Along all the process it is clear many differences between the two models. The differences are related to the behavior of the software in relationship to the use, structure, limitations, data import and speed of the queries. Figure 3 show the import times in these two models.

According to the graphics present in Figure 2 and Figure 3 it is possible to see that in relation to the data import the performance of the DW was not so good, being 7% less efficient than the non-relational model. However, in relationship to the queries, the multidimensional model had a better performance, using only 36% of the time used in the Hbase. The weak point in the DW was imported the fact table because the operational base has tables with more than 14 million lines as *imposto_item_nf*. In relation to the queries, it was noticed that the non-relational model is not so adequate when using many joins between the tables. About storage, the two models are similar in performance. Regarding flexibility, it was observed that the HBase was better. It is due to fact that the HBase is a non-relational database, oriented to columns, that received a big amount of structured data adapting them in a column structure and through the Hive integrate this data and make join in the tables.

There are two important points in this comparison. Although the multidimensional model had a good performance in the queries,

Table 5. : Summary of consumed time and total of lines returned in each query for the two models

Model	Query 1		Query 2		Query 3	
	Lines	Time	Lines	Time	Lines	Time
Multidimensional	222	2min 9s	110	1min 6s	1940	1min 32s
Non relational	441.912	5min 58s	441.498	3min 36s	256.832	3min 45s

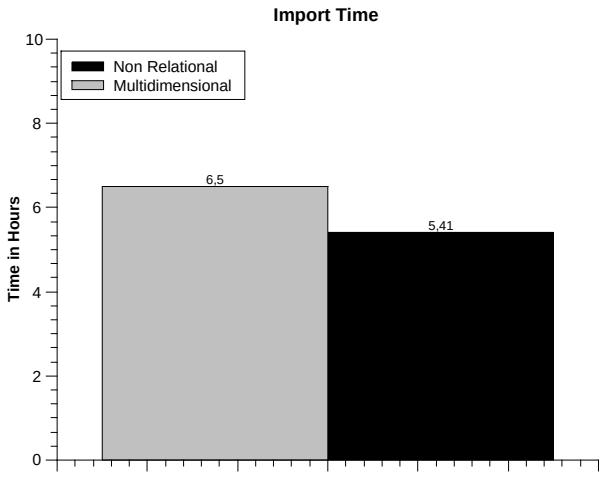

Fig. 3: Comparison about the import time in the two studied models

the OLAP tool could not process the queries evolving many dimension tables with a great volume of data, as *dim_produto* and *dim_destinatario*. At the moment that the queries were running the tool lost the connection with the database and could not finish the process. In relation to the HBase, it runs in the Hadoop framework, which means that the process can be distributed. Also, the framework is very flexible to configure many items, for example, change physical or virtual memory, adding a node in the cluster, and others. The Hadoop framework is very complete and has a lot of tools that can help a lot of the user.

5. CONCLUSION

In this paper was realized a study looking to comparing two different data models in the context of a big volume of data. In order to make this comparison was used the Data Warehouse with the multidimensional model and the NoSql database-oriented to the column with the non-relational model. The Hbase was selected as the NoSQL database because it was a model oriented to columns which allow access to just one column (and not all set) with many lines at the same time, it is one of the main open-source NoSQL systems and aleatory access of reading and writing in real-time using commodity hardware.
With relation to the difficulties in the experiment, the fact table took almost 300 hours to be loaded (summed all attempts to load it). The Hadoop was installed in Xubuntu 14 because it was presenting technical problems to be installed in Windows 10. The standard tool to import the data from the relational database to the Hbase was the Scoop. However, the system always runs out of memory when the table which was being imported had more than 600.000 lines. Therefore, was used the Hadoop tool ImportTsv.

Finally, work with big volumes of data is not trivial, it requires a lot of studies to choose the best approach to the problem. It was observed that the Hbase did not have a so good performance with the queries. The point is that the fact table in the multidimensional model, where all the quantitative data is already there, make the queries much faster. In the non-relational model oriented to the column, the joins always need to be connected in two big tables (*nfe* and *item_nf*), and in each query that seeks the total amount of sales or units sold it is necessary to make a sum operation for each one, reducing the performance of the query. Besides that, the Hbase is an architecture to the distributed system, so if the experiment was conducted in many nodes in a cluster instead of a single machine the performance of the non-relational model probably could be better. Therefore, in this context, where data obtained from sales operations were transformed from XML to structured data, the multidimensional model was superior to the non-relational model, this is due to the fact that this kind of experiment was required a huge quantity of complex queries and analysis of routine data.

6. REFERENCES

[1] ALLIANCE. Whats the deal big deal with data? Technical report, BSA, 20 F Street, NW, Washington, 2015.

[2] Andrew McAfee, Erik Brynjolfsson, et al. Big data: the management revolution. *Harvard business review*, 90(10):60–68, 2012.

[3] John Gantz and David Reinsel. Extracting value from chaos. *IDC iview*, 1142:1–12, 2011.

[4] Lei Wang, Jianfeng Zhan, Chunjie Luo, Yuqing Zhu, Qiang Yang, Yongqiang He, Wanling Gao, Zhen Jia, Yingjie Shi, Shujie Zhang, et al. Bigdatabench: A big data benchmark suite from internet services. In *High Performance Computer Architecture (HPCA), 2014 IEEE 20th International Symposium on*, pages 488–499. IEEE, 2014.

[5] Jason Richmond and Jinhua Guo. Pricing the internet for congestion control and social welfare. In *Computer Communication and Networks (ICCCN), 2014 23rd International Conference on*, pages 1–6. IEEE, 2014.

[6] Ibrahim Abaker Targio Hashem, Ibrar Yaqoob, Nor Badrul Anuar, Salimah Mokhtar, Abdullah Gani, and Samee Ullah Khan. The rise of big data on cloud computing: Review and open research issues. *Information Systems*, 47:98–115, 2015.

[7] Seref Sagiroglu and Duygu Sinanc. Big data: A review. In *Collaboration Technologies and Systems (CTS), 2013 International Conference on*, pages 42–47. IEEE, 2013.

[8] Josiane Rodrigues, Marco Cristo, Javier Zambrano Ferreira, David Fernandes, and André Carvalho. Multi-entity polarity analysis and detection of subjectivity in financial documents. *Journal of Information and Data Management*, 6(2):130, 2016.

[9] Sam Madden. From databases to big data. *IEEE Internet Computing*, 16(3):4–6, 2012.

[10] Jing Han, E Haihong, Guan Le, and Jian Du. Survey on nosql database. In *Pervasive computing and applications (ICPCA), 6th international conference on*, pages 363–366. IEEE, 2011.

[11] Hsinchun Chen, Roger HL Chiang, and Veda C Storey. Business intelligence and analytics: From big data to big impact. *MIS quarterly*, 36(4):1165–1188, 2012.

[12] Paul Lane and Oracle9i Data Warehousing Guide. Release 1 (9.0. 1). *Oracle Corporation*, 2001.

[13] Marcos Rodrigues Vieira, J Figueiredo, Gustavo Liberatti, and Alvaro Fellipe Mendes Viebrantz. Nosql databases: Concepts, tools, languages and case study in a big data context. *Brasilian Database Congress*, 2012.

[14] Ling Liu and M Tamer Özsu. *Encyclopedia of database systems*, volume 6. Springer Berlin, Heidelberg, Germany, 2009.

[15] Manuel Serrano, Coral Calero, and Mario Piattini. Experimental validation of multidimensional data models metrics. In *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*, page 327. IEEE, 2003.

[16] Torben Bach Pedersen and Christian S Jensen. Multidimensional data modeling for complex data. In *Data Engineering. Proceedings., 15th International Conference on*, pages 336–345. IEEE, 1999.

[17] Ralph Kimball and Margy Ross. *The data warehouse toolkit: the complete guide to dimensional modeling*. John Wiley & Sons, 2011.

[18] William Rowen, Il-Yeol Song, Carl Medsker, and Edward Ewen. An analysis of many-to-many relationships between fact and dimension tables in dimensional modeling. In *Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW 2001), Interlaken, Switzerland*, 2001.

[19] William H Inmon. What is a data warehouse? *Prism Tech Topic*, 1. (1995), 1995.

[20] William H Inmon. *Building the data warehouse*. John wiley & sons, 2005.

[21] John Palo and Newman Ray. Creation and adoption of online analytical process (olap) into the management decision support system aided by computers. *Scholedge International Journal of Business Policy & Governance ISSN 2394-3351*, 2(5):8–13, 2015.

[22] María Carina Roldán. *Pentaho Data Integration Beginner's Guide*. Packt Publishing Ltd, 2013.

[23] DataOnFocus. Oltp vs olap: De nitions and comparison, 2015.

[24] Ricardo Brito. Nosql databases x relational databases: comparative analysis. Technical report, Fortaleza University, Av. Washington Soares, Fortaleza-CE, 2010.

[25] Rick Cattell. Scalable sql and nosql data stores. *Acm Sigmod Record*, 39(4):12–27, 2011.

[26] IBM. Bringing big data to the enterprise, 2016.

[27] Eric A Brewer. Towards robust distributed systems. In *PODC*, volume 7, 2000.

[28] C. Strauch, U.-L. S. Sites, and W. Kriha. Nosql databases, 2016.

[29] ABM Moniruzzaman and Syed Akhter Hossain. Nosql database: New era of databases for big data analytics-classification, characteristics and comparison. *International Journal of Database Theory and Application*, 6(4):1–14, 2013.

[30] Berk Atikoglu, Yuehai Xu, Eitan Frachtenberg, Song Jiang, and Mike Paleczny. Workload analysis of a large-scale key-value store. In *ACM SIGMETRICS Performance Evaluation Review*, volume 40, pages 53–64. ACM, 2012.

[31] Daniel J Abadi, Peter A Boncz, and Stavros Harizopoulos. Column-oriented database systems. *Proceedings of the VLDB Endowment*, 2(2):1664–1665, 2009.

[32] Ameya Nayak, Anil Poriya, and Dikshay Poojary. Type of nosql databases and its comparison with relational databases. *International Journal of Applied Information Systems*, 5(4):16–19, 2013.

[33] Kristina Chodorow. *MongoDB: the definitive guide*. O'Reilly Media, Inc., 2013.

[34] Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall, and Werner Vogels. Dynamo: amazon's highly available key-value store. *ACM SIGOPS operating systems review*, 41(6):205–220, 2007.

[35] Ricardo Neves and Jorge Bernardino. Performance and scalability of voldemort nosql. In *Information Systems and Technologies (CISTI), 2015 10th Iberian Conference on*, pages 1–6. IEEE, 2015.

[36] Avinash Lakshman and Prashant Malik. Cassandra: structured storage system on a p2p network. In *Proceedings of the 28th ACM symposium on Principles of distributed computing*, pages 5–5. ACM, 2009.

[37] Brad Fitzpatrick. Distributed caching with memcached. *Linux journal*, (124):5, 2004.

[38] Lars George. *HBase: The Definitive Guide: Random Access to Your Planet-Size Data*. O'Reilly Media, Inc., 2011.

[39] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The hadoop distributed file system. In *Mass storage systems and technologies (MSST), 2010 IEEE 26th symposium on*, pages 1–10. IEEE, 2010.

[40] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff, and Raghotham Murthy. Hive: a warehousing solution over a map-reduce framework. *Proceedings of the VLDB Endowment*, 2(2):1626–1629, 2009.

[41] John M Zelle. *Python programming: an introduction to computer science*. Franklin, Beedle & Associates, Inc., 2004.

[42] Ralph Kimball and Margy Ross. *The data warehouse toolkit: The definitive guide to dimensional modeling*. John Wiley & Sons, 2013.