An Architecture for Data Warehousing in Big Data Environments

Bruno Martinho, Maribel Yasmina Santos

An Architecture for Data Warehousing in Big Data Environments

Bruno Martinho and Maribel Yasmina Santos

ALGORITMI Research Centre, University of Minho, Guimarães, Portugal
bruno_martinho@dsi.uminho.pt, maribel@dsi.uminho.pt

Abstract. Recent advances in Information Technologies facilitate the increasing capacity to collect and store data, being the Big Data term often mentioned. In this context, many challenges need to be addressed, being Data Warehousing one of them. In this sense, the main purpose of this work is to propose an architecture for Data Warehousing in Big Data, taking as input a data source stored in a traditional Data Warehouse, which is transformed into a Data Warehouse in Hive. Before proposing and implementing the architecture, a benchmark was conducted to verify the processing times of Hive and Impala, understanding how these technologies could be integrated in an architecture where Hive plays the role of a Data Warehouse and Impala is the driving force for the analysis and visualization of data. After the proposal of the architecture, it was implemented using tools like the Hadoop ecosystem, Talend and Tableau, and validated using a data set with more than 100 million records, obtaining satisfactory results in terms of processing times.

Keywords: Big Data · Data Warehouse · NoSQL · Hadoop · Hive · Impala

1 Introduction

Nowadays, due to the high competitiveness that exists between organizations, they need to invest more and more in technology. Usually, the cause of this need involves the frequent change of the business trends as well as their customers' habits [1]. Data Warehouse and On-line Analytical Processing (OLAP) are technologies that have been following this evolution to the present day [1], being a Data Warehouse a database to support analytical processing and to assist in decision making process [2]. The implementation of these systems usually occurs in relational databases that may not be able to store and process large volumes of data [3].

With the recent technological advances, organizations are collecting more and more data, with different types, formats and speeds. When used and analyzed in the proper way these data have enormous potential, enabling organizations to completely change their business systems for better results [4]. Transforming the potential of the information, in this increasingly digital world, requires not only new data analysis algorithms, but also a new generation of systems and distributed computing environments to deal with the sharp increase in the volume of data and its lack of structure [5]. The challenge is to enhance the value of these data, as these are sometimes in

completely different formats [6]. Combining the large amounts of data with the need to analyze them, there is a need to think the role of Data Warehousing in the context of Big Data, being Big Data the ability to collect, store and process large volumes of data [4]. Big Data refers mainly to the massive amounts of unstructured data produced by high-performance applications [7], but also data that arrive in structured and semi-structured formats [8]. The Big Data solutions are ideal for data analysis from a variety of sources [9], which characteristics (volume, velocity, variety) make Big Data a major challenge for organizations that still use the traditional mechanisms for data management.

Given this context, the question is where to store these massive amounts of data for analytical purposes and the data models that must be used. Regarding storage, the movement called NoSQL (Not Only SQL) promotes many innovative solutions for the storage and processing of large volumes of data [10]. These databases, usually, do not provide any guidelines on how to model and implement a Data Warehouse for Big Data contexts. Regarding specific technologies, Hadoop Data File System (HDFS) and Hive [11] for storage and Impala [12] for processing are frequently mentioned.

So far, the development of Data Warehousing in Big Data was guided by use-case driven approaches in which specific technologies and implementation contexts are proposed and tested to solve some specific problems [13]. Although these relevant works provide useful guidelines on how to proceed, they do not envisage the proposal of a generic architecture that complement storage and processing technologies for the efficient implementation of a Big Data Warehouse.

For achieving this aim, this work benchmarks Hive and Impala for data processing, while Hive is used as the Big Data Warehouse repository [12], [14]. The knowledge obtained from the performed benchmark is crucial for proposing an architecture for the implementation of Big Data Warehouses, which was tested in a demonstration case that stored and processed more that 100 million records.

This paper is organized as follows. Section 2 presents the related work. Section 3 summarizes the main findings in the benchmark performed to compare the performance of Hive and Impala for data processing. Section 4 describes the proposed architecture, while section 5 shows some results from its implementation. Section 6 concludes with some remarks and guidelines for future work.

2 Related Work

Being Big Data a recent research topic, there is no common approach on how to design and implement a Big Data Warehouse. Many authors discuss this need and propose works that are mainly guided by use-case driven approaches, where specific solutions are recommended and tested, mainly giving non-structured guidelines on how to design Big Data Warehouses and, mostly, revisiting traditional modeling techniques. However, the traditional logical data models used in the implementation of Data Warehouses do not fit into a featured environment of large amounts of data as in Big Data and, so, repositories like NoSQL and Hadoop are the most recent bets in data storage [2], providing infrastructures for implementing Data Warehouses and

multidimensional data structures in the form of OLAP data cubes [15]. In the work of [4], the need for redesigning traditional Data Warehouses, in order to address new challenges like data types, data volume, user requirements and performance is focused. Moreover, this author mentions that Big Data Warehouses need to include data from several sources and must be implemented making use of multiple technologies like relational database management systems, Hadoop, NoSQL databases, reporting and visualization, among others.

From the technological point of view, many technologies have been proposed, mainly in what concerns storage, being NoSQL databases the most noticeable example, with more than 225 NoSQL databases already proposed, as reported in http://nosql-database.org. From the data modeling point of view, very specific approaches have been followed, mainly driven by very specific data requirements scenarios. In NoSQL databases, as logical data models are schema-free, meaning that different rows in a table may have different data columns (less rigid structures) or that the defined schema may change on runtime, the definition of data schemas follows a different approach [16]. Instead of reflecting the relevant entities in a particular domain and the relationships between those entities, data schemas are defined considering the queries that need to be answered, being data replicated as many times as needed [17], given the importance of query performance when huge volumes of data are being processed [18].

The transformation of traditional data models into data models for NoSQL databases mainly follows two types of repositories, based on columns and on documents. The authors in [2,3] propose an approach for mapping a conceptual model of a traditional data environment into a logical data model in HBase and MongoDB for data storage in a distributed environment [19]. In these works, the authors use columns and documents oriented databases as data storage areas without the integration of Hive. In this sense, and as the Hive is considered the Data Warehouse in the context of Big Data, because of its analytical operators, the databases used by those authors do not make available different analytical perspectives on the data.

In another work, [18], the authors recognize that the design of big data warehouses is very different from traditional data warehouses, as their schema should be based on novel logical models allowing more flexibility than the relational model does. The authors propose a design methodology for the representation of a multidimensional schema at the logical level based on the key-value model. In this approach, a data-driven approach design, using data repositories as main source of information, and a requirements-driven approach design, using information of the decision makers, are integrated.

Given this overall context, this work proposes an architecture that uses the Hadoop Data File Systems (HDFS) as staging area and Hive as a Data Warehouse. For defining the Data Warehouse logical model, a set transformation rules are used [13], deriving a tabular data model for Hive, taking into consideration a multidimensional data model with the data requirements for a specific data analytics scenario. These rules [13] provide as output a set of tables with different analytical perspectives on data, imitating the on-line analytical processing cubes normally used in traditional Business Intelligence environments.

In order to improve the performance of the proposed architecture, both in the ETL/ELT and in the analysis and visualization of data, a benchmark was performed to verify how Hive and Impala perform. Impala is tested to verify how it performs when analyzing data that is stored in Hive. According to [20], Impala is faster in querying the data when compared to Hive, as it uses a query engine that does not need MapReduce [20,21] and, as Hive uses MapReduce jobs, its performance is slower than the performance of Impala [21].

However, in some of these scenarios, where Impala and Hive are compared, the performance of Impala was not analyzed with the data stored in Hive, like this work proposes, where Impala is only acting as a querying mechanism, and not as a data storage repository with tables that enhance data analytics over different perspectives. In this work, the performed benchmark does not use a simple table with columns in the Hive, but organizes the data also using partitions and buckets, stored in the parquet format and using the snappy compression. In this sense, besides verifying the performance of Hive and Impala, another objective was to verify if Impala is able to use the same data types as Hive, if interprets the partitions and buckets correctly and, also, if uses the compression formats. Summarizing, the objective of the benchmark was not only to verify performance, but also the combination of these two technologies, as they can be used as complementary technologies instead of competitors in querying data.

3 Benchmarking Hive and Impala

The study of technologies that can be integrated with Hive, allowing better querying performance, is important in this work, and Impala emerged in this direction. According to [21], Impala emerged as an addition for querying Hive tables, which can be faster than the Hive mechanism itself. Also, the authors mention that Hive is more suitable for storing and processing large amounts of data in batch and Impala for processing in real time. In this work, Hive remains as a data storage mechanism in the form of Data Warehousing and uses its querying component for the creation, aggregation and transformation of data for the Hive tables itself, being more convenient for ELT processes than Impala [20]. Once the tables are created and the data is stored, Impala is used as a query engine to analyze the information in several dashboards.

The integration between Hive and Impala can be achieved through the use of the metastore, where all metadata associated to the Hive tables is stored [22]. Given this context, **Fig. 1** shows how these two technologies can be integrated, having as implementation environment the Hadoop ecosystem. Considering the benchmarks already mentioned, an appropriate solution suggests the integration these two technologies, taking advantage of their characteristics, compatibility and performance.

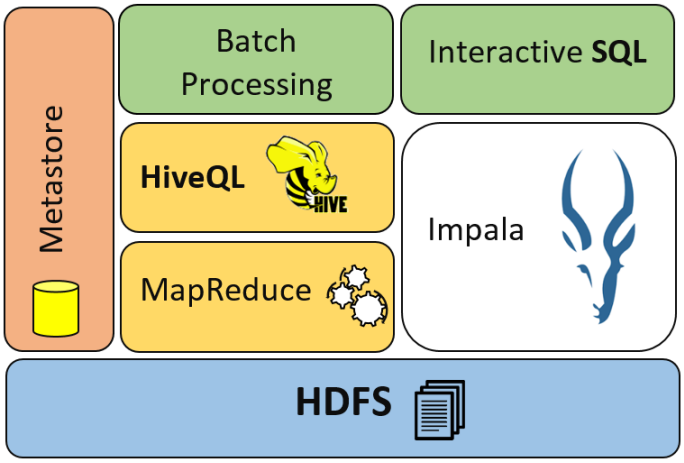

Fig. 1. Integration between Hive and Impala (Source: Adapted from [22])

In order to achieve better performance when processing Hive tables, those must be stored in the parquet format [20]. This is a columnar format that best suits querying either in Hive and Impala, taking into consideration CPU and memory consumption [20]. Moreover, the snappy compression method can be used for reducing the size of the data by half or more, relieving the IO pressure [20].

For understanding the contexts in which Impala querying processing is better than Hive querying processing, for data stored in Hive, a set of queries were defined and executed. The dataset used in this benchmark includes more than 100 million records associated with flights in the USA [23]. The used technological infrastructure includes a virtual machine Intel Core i5-2430 CPU 240 GHz with the Centos 6.4 operating system, 6 GB of RAM and 100 GB SSD. This machine has Hadoop based on a single node cluster.

Table 1 shows the five queries that were defined to test the performance of Impala and Hive. Those are single queries over a table, including some aggregation functions and selection or grouping conditions. The dataset used in this benchmark is better explained in section 4.2.

Table 1. Defined queries for the benchmark

Query 1	SELECT Carrier name, SUM (Total delay) as Total delays FROM Delays WHERE Year = '2008' GROUP BY Carrier name;
Query 2	SELECT Carrier name, Year, SUM (Accomplished flight) as Number of flights FROM Delays GROUP BY Carrier name, Year ORDER BY Carrier name, Year;
Query 3	SELECT Time interval, AVG (Delay in departure) as Minutes delay departures FROM Delays WHERE Year = '2007' GROUP BY Time interval;
Query 4	SELECT State to, Month, COUNT (Accomplished flight) as Number of flights FROM Delays WHERE Year = '2008' GROUP BY State from, State to, Month;
Query 5	SELECT DISTINCT Carrier name FROM Delays;

The obtained results, in terms of processing times, for the five queries presented in **Table** 1, are shown in **Fig. 2**. As can be seen, Impala had better results when compared with Hive in querying the data. The difference was more than 30 seconds in each query, which represents a significant improvement. For setting the needed processing time, each query was run three times, and the average of these three results was taken as the time that is shown in **Fig. 2**.

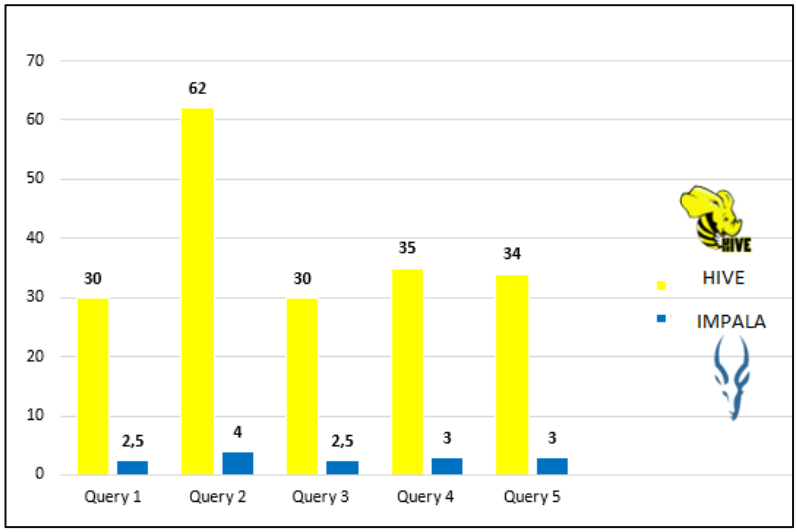

Fig. 2. Benchmark of Hive and Impala (time in seconds)

4 An Architecture for Big Data Warehousing

4.1 Overall Overview

The proposed architecture makes use of multiple technologies such as HDFS for storing facts and dimension tables in different files (staging area); Hive to act as a Data Warehouse, containing the final data set for the data analytics and visualization tasks; Impala for querying the Hive tables (giving the results of the performed benchmark); Talend Open Studio for Big Data, which is responsible for all data flows and ETL/ELT (Extract, Transform, Load)/(Extract, Load, Transform) processes; and, finally, Tableau (www.tableau.com) as the tool for the implementation of analytical dashboards.

As can be seen in **Fig. 3**, this architecture considers that an organization can have a traditional Data Warehousing environment that needs to be migrated to a Big Data environment, or that this Data Warehouse does not exist. In this case, the operational data sources can be used to feed the Big Data Stating Area, which will support the loading of the Big Data Warehouse.

In this work, we are considering that a traditional Data Warehouse exist, showing how organizations can move to a Big Data context using the organizational knowledge and the corresponding logical data models that guided the concretization of an analytical environment. Although this is not mandatory, this approach helps in setting the logical data model for the Big Data Warehouse, as all the data requirements are available in the multidimensional data model of the traditional Data Warehouse.

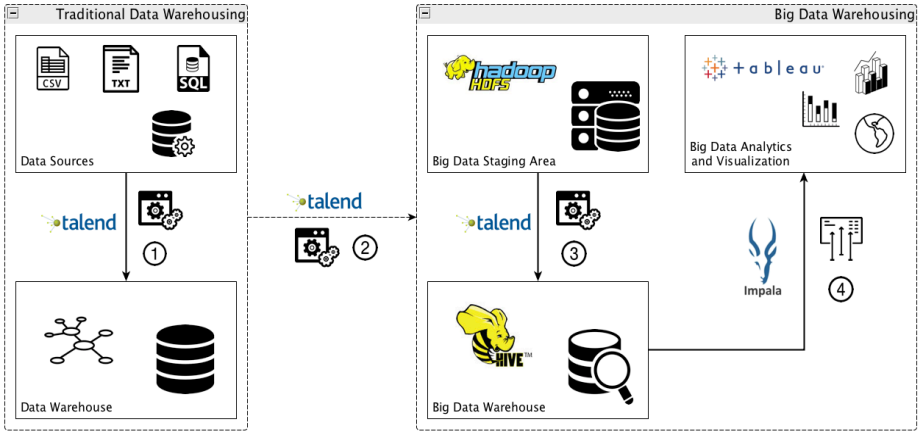

Fig. 3. An Architecture for Data Warehousing in Big Data

The analysis of **Fig. 3** shows the several considered components, already mentioned, and the data flows among them. The **1st data flow** consists in the ETL of operational data sources (that can be in different formats) to the traditional Data Warehouse, considering the defined logical data model (with the dimension and fact tables). The **2nd data flow** includes the ELT of dimension and facts tables, stored in the traditional Data Warehouse, to HDFS (where each table is stored in a different file).

In case the traditional Data Warehouse does not exist, the operational data sources can follow the same path, being stored in HDFS in different files. HDFS is used as a staging area in the implementation of the Big Data Warehouse in Hive. The **3rd data flow** is needed for feeding the Hive tables in an ELT process that stores each file present in HDFS in a Hive table. In the scenario recommended in this work, each file corresponds to a dimension or fact table. Once in Hive, these tables are used to perform a set of transformations that leads to new tables, optimized for query processing, as they integrate dimension and fact tables in a way that imitate an analytical cube for analysis and visualization tasks in decision making contexts. These transformations are better explained in the following subsection.

The **4th data flow** is the querying engine that uses Impala for querying the Hive tables, feeding the analytical dashboards used in data analysis and visualization tasks. In technological terms, Tableau is connected to Impala, which interacts with the Hive metastore, querying the data available in the Big Data Warehouse.

4.2 Logical Data Model for a Big Data Warehouse

In Big Data Contexts, logical data models are usually defined attending to the queries that need to be answered. In this work, we use the proposal of [13] for setting the logical data model of a Big Data Warehouse, as the authors propose a set of rules that automatically transform a multidimensional data model in a tabular model suited to be implemented in Hive. This approach has as advantage the use of the data and analyti-

cal requirements identified in the multidimensional data model, guiding the process of implementation of a Big Data Warehouse.

The approach proposed by [13] allows the identification of a complete set of tables that imitate the way how analytical data cubes perform in traditional Business Intelligence contexts. The approach combines different dimension and fact tables, integrating them in Hive tables, providing the different analytical perspectives.

As it is not possible (also, it is not the objective) in this paper to explain all these rules in detail, and show all the transformations, this work uses two of the obtained Hive tables for demonstration purposes. The transformation process was started taking into consideration a multidimensional model that includes 7 dimension tables (*Calendar*, *Time*, *Airport*, *Flight*, *Carrier*, *Airplane*, *Cancellation*) and two fact tables (*Flights*, *Delays*), for storing and analyzing data about commercial flights in the USA. Following the rules in [13], a set of 127 different Hive tables can be identified, being able to answer all possible questions for this dataset. These tables include different data granularities, meaning different levels of detail, being possible to choose the appropriate ones for specific analytical contexts or choose some of the more detailed ones and use them to obtain more aggregated data. This is possible when analytical tools like Tableau are used for analyzing the more detailed Hive tables, as the defined aggregation functions will allow data summarization.

Without giving too much detail, **Fig. 4** briefly shows how the combination of dimension and fact tables can be achieved in the transformation process to derive the Hive tables. As can be understood, different combinations would lead to different Hive tables, either in terms of the available attributes for data analysis, either in the level of detail of each table.

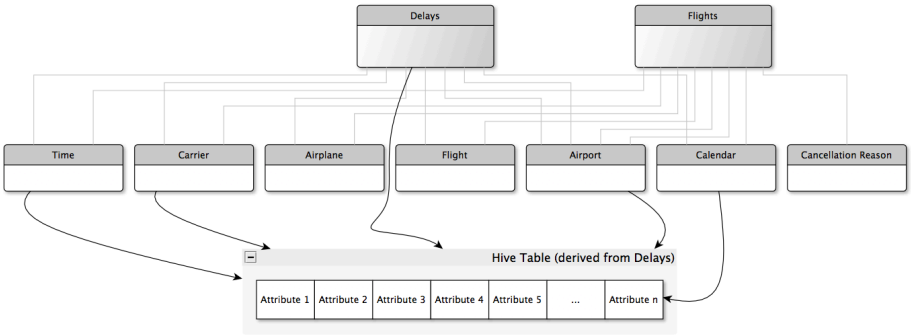

Fig. 4. Transformation process for driving the Hive tables

From this set of tables, **Fig. 5** and **Fig. 6** show how the two that were selected are constituted, in terms of columns. The columns can be descriptive (those that are inherited from dimension tables) and analytical (those that are inherited from fact tables). Due to the extensive number of business indicators available in the multidimensional data model used as source of the data and analytical requirements, both figures present only a subset of the available metrics.

The *aggDelays* table characterizes the delays of the several flights, considering the several locations in terms of airports of departure and arrival, the carrier, the calendar

dimension, and the time of the day in which the flight took place. As business indicators, the total delay in minutes, the delay considering several reasons (security, company, weather, ...) and the number of delayed flights, among other metrics, are considered. The time component is divided in time intervals ([00:00 - 01:00[, [01:00 - 02:00[, and so on).

As can be seen in **Fig. 5**, the analytical columns derived from the fact table include, for each attribute, an aggregation function that can allow the summarization of the dataset (this will depend on the level of detail of each descriptive column regarding the analytical columns).

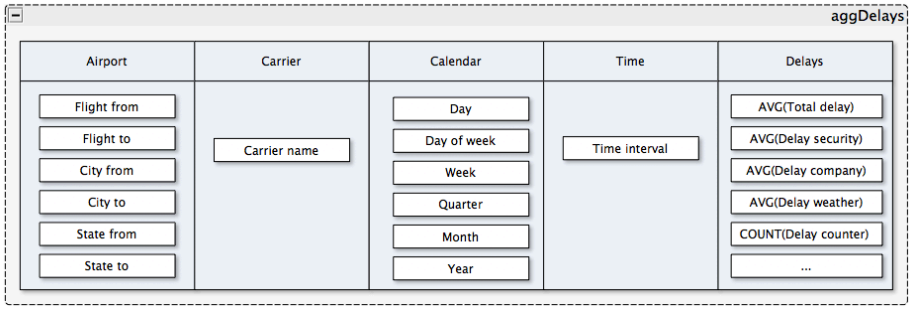

Fig. 5. *aggDelays* Hive table of the Big Data Warehouse

The *aggFlights* table (**Fig. 6**) includes the information about all the flights, but here aggregated attending to the airports (origin and destination), the carrier, the airplane and the cancellation type (in case of flight cancellation). As business indicators the duration of the flights, the traveled distance, the number of flights, among others attributes, can be analyzed.

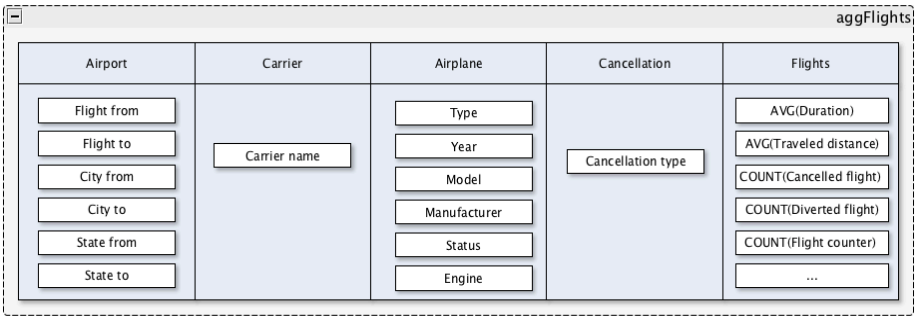

Fig. 6. *aggFlights* Hive table of the Big Data Warehouse

The number of records in each table depends on the level of aggregation considered in the transformation process, which varies attending to the dimensions that are combined for a specific fact table. More or less detailed tables can be obtained. In the examples here considered, the first table (*aggDelays*) has 123 534 969 records, con-

taining all the available data as no summarization was here possible, while the second table (*aggFlights*) has 3 774 583 records, accomplishing a significant summarization of the available data.

5 Demonstration Case

In the implementation of the proposed architecture, the technologies previously shown in **Fig. 3** were used. In the demonstration case, a traditional Data Warehouse containing more than 100 million records was used as the data source, related to data about flights in the USA [23]. According to the architecture, the available data was extracted from the traditional Data Warehouse and stored in HDFS, creating one file per table. After this process, the data were transferred to Hive, where the logical data model was identified and implemented, attending to the transformation rules specified in [13]. Once the Big Data Warehouse was loaded, Impala was used as the query engine, allowing data analytics over the available data.

Tableau was used as the front-end tool, where specific dashboards were implemented. As an example of the analytical tools that can be provided to the users, **Fig. 7** shows a dual-axis plot with the number of flights per time interval (blue bars) and the average delay per flight in minutes (red line). In this plot, it is possible to see the hours in which more flights were verified. Also, it calls our attention that it is at the end of the day, late afternoon, that the highest average delay value is verified, around 30 minutes, in the time interval [19:00 - 20:00[.

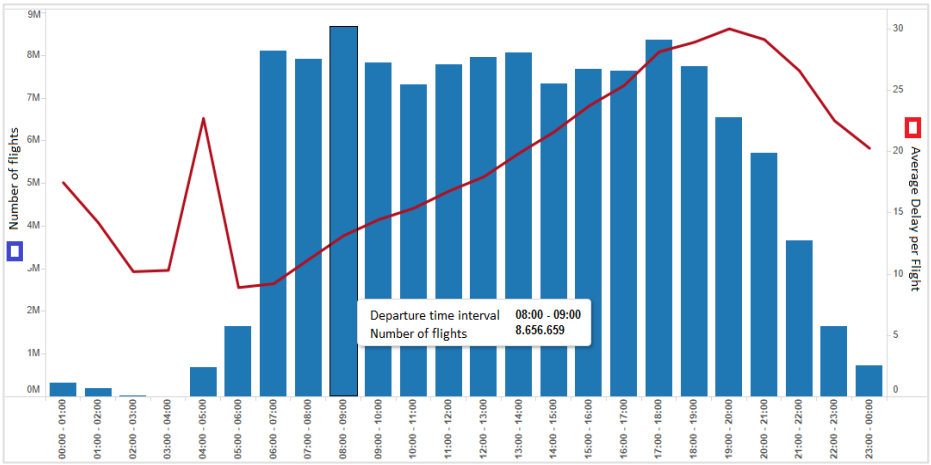

Fig. 7. Number of flights and average delay per flight

Another example of the analytical capabilities that can be provided to the users is shown in the map of **Fig. 8**, where a color scale highlights the states with higher incidence of flights, with Texas (TX) and California (CA) each having more than 11% of the total number of flights.

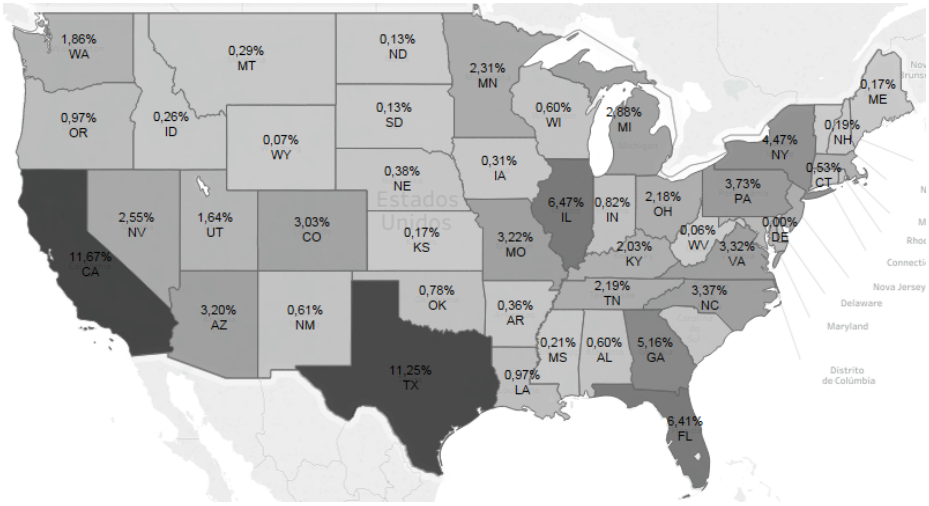

Fig. 8. Percentage of the number of flights per state

To verify if, using Tableau as front-end, the processing times are adequate for decision support tasks, the time needed for processing several queries that integrate the implemented dashboards, already presented in **Table 1**, was analyzed. Once again, and although the architecture proposes Impala as the query engine, the processing times needed by Hive and by Impala were compared. The results are shown in **Fig. 9**.

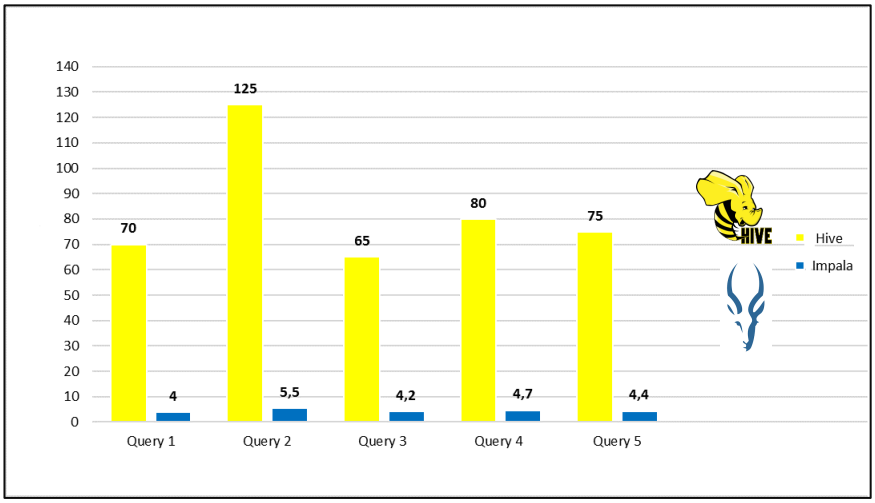

Fig. 9. Benchmark of Hive and Impala through Tableau (time in seconds)

As can see in **Fig. 9**, Impala maintains the good performance shown before, for processing the queries without Tableau as front-end, while Hive clearly decreases its

performance, obtaining processing times higher than 120 seconds, a scenario that is not satisfactory in an iterative analytical context.

6 Conclusions

This paper presented an architecture for implementing Big Data Warehouses, which uses HDFS as the staging area, Hive as the Data Warehouse, Impala as the query engine and Tableau as the front-end analytical tool. For all extraction, loading and transformation activities, Talend Open Studio for Big data was used. Using as data source a traditional Data Warehouse, the concretization of a demonstration case allowed the integration of all the proposed components and technologies, providing an analytical environment where Impala ensures satisfactory processing times.

In this architecture, and for the specification of the data requirements, particular attention was given to the Hive data model, which was derived considering a multidimensional data model of a traditional Data Warehouse.

As future work, the architecture will be extended for considering real-time processing needs, both in feeding the Big Data Warehouse and in the interactive analysis and visualization of the data.

Acknowledgments. This work has been supported by COMPETE: POCI-01-0145-FEDER-007043 and FCT (*Fundação para a Ciência e Tecnologia*) within the Project Scope: UID/CEC/00319/2013, and by Portugal Incentive System for Research and Technological Development, Project in co-promotion nº 002814/2015 (iFACTORY 2015-2018). Some of the figures in this paper use icons made by Freepik, from www.flaticon.com.

References

1. Santhosh, B., Renjith, K.: Next Generation Data Warehouse Design with OLTP and OLAP Systems Sharing same Database. International Journal of Computer Applications, **72** (13), 45-50 (2013). doi:10.5120/12557-9282
2. Dehdouh, K., Bentayeb, F., Boussaid, O., Kabachi, N.: Using the column oriented NoSQL model for implementing big data warehouses. In: International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA), Athens, 8-11 September (2015)
3. Chevalier, M., Malki, M.E., Kopliku, A., Teste, O., Tournier, R.: Implementing Multidimensional Data Warehouses into NoSQL. In: The 17th International Conference on Entreprise Information Systems (ICEIS), Barcelona, Spain, (2015)
4. Krishnan, K.: Data warehousing in the Age of Big Data, 1st Edition, Morgan Kaufmann, Elsevier Inc., (2013)
5. Aye, K.N., Thein, N.L.: A Comparison of Big Data Analytics Approaches Based on HadoopMapReduce. In: The 11th International Conference on Computer Applications, Yangon, Myanmar, (2013)
6. Khan, M.A.-u.-d., Uddin, M.F., Gupta, N.: Seven V's of Big Data Understanding Big Data to extract Value. In: Zone 1 Conference of the American Society for Engineering Education, Bridgeport, CT, 3-5 April (2014)

7. Cuzzocrea, A., Song, I., Davis, K.: Analytics over large-scale multidimensional data: the big data revolution. In: The ACM 14th International Workshop on Data Warehousing and OLAP, New York, USA,
8. Colin, W.: Using Big Data for Smarter Decision Making. IBM White Papers & Reports, (2011)
9. Zikopoulos, P., Eaton, C., deRoos, D., Deutsch, T., Lapis, G.: Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data, 1 ed. McGraw-Hill, (2011)
10. Chevalier, M., El Malki, M., Kopliku, A., Teste, O., Tournier, R.: Implementation of multidimensional databases with document-oriented NoSQL. In: International Conference on Big Data Analytics and Knowledge Discovery, 379–390 (2015)
11. Thusoo, A., Sarma, J.S., Jain, N., Shao, Z., Chakka, P., Zhang, N., Antony, S., Liu, H., Murthy, R.: Hive – A Petabyte Scale Data Warehouse UsingHadoop. In: IEEE 26th International Conference on Data Engineering (ICDE), Long Beach, CA, 1-6 March (2010)
12. Kornacker, M., Behm, A., Bittorf, V., Bobrovytsky, T., Ching, C., Choi, A., Erickson, J., Grund, M., Hecht, D., Jacobs, M., Joshi, I., Kuff, L., Kumar, D., Leblang, A., Li, N., Pandis, I., Robinson, H., Rorke, D., Rus, S., Russell, J., Tsirogiannis, D., Wanderman-Milne, S., Yoder, M.: Impala: A Modern Open-Source SQL Engine for Hadoop. Cloudera, (2015)
13. Santos, M.Y., Carlos, C.: Data Warehousing in Big Data: From Multidimensional to Tabular Data Models. In: The International Conference on Computer Science & Software Engineering, Porto, 20-22 July (2016)
14. Dhawan, S., Rathee, S.: Big Data Analytics using Hadoop Components like Pig and Hive. American International Journal of Research in Science, Technology, Engineering & Mathematics, 88-93 (2013)
15. Dehdouh, K., Bentayeb, F., Boussaid, O., Kabachi, N.: Columnar NoSQL CUBE: Agregation operator for columnar NoSQL Data Warehouse. In: The IEEE International Conference on Systems, Man and Cybernetics (SMC), San Diego, CA, 5-8 Oct, (2014)
16. Santos, M.Y., Carlos, C.: Data Models in NoSQL Databases for Big Data Contexts. In: The International Conference on Data Mining and Big Data, Indonesia, Bali, (2016)
17. Tamás, V., Péter, F., Krisztián, F., Hassan, C.: Denormalizing data into schema-free databases. In: IEEE 4th International Conference on the Cognitive Infocommunications (CogInfoCom), Budapest, 2-5 Dec, (2013)
18. Tria, F., Lefons, E., Filippo, T.: Design process for Big Data Warehouses. In: The International Conference on Data Science and Advanced Analytics (DSAA), Shanghai, Oct. 30 -Nov.1 (2014)
19. Dan, H., Eleni, S.: A Three-Dimensional Data Model in HBase for Large Time-Series Dataset Analysis. In: The IEEE 6th International Workshop on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems, Trento, Italy (2012)
20. Xiaopeng, L., Zhou, W.: Performance Comparison of Hive, Impala and Spark SQL. In: The 7th International Conference on Intelligent Human-Machine Systems and Cybernetics, Hangzhou, China, 26-27 Aug (2015)
21. Li, J.: Design of real-time data analysis system based on Impala. In: The Advanced Research and Technology in Industry Applications, Ottawa, Canada, 29-30 Sept (2014)
22. Kulkarni, K., Lu, X., Panda, D.K.: Characterizing Cloudera Impala Workloads with BigDataBench on InfiniBand Clusters. In: The 7th Workshop on Big Data Benchmarks, Performance, Optimization, and Emerging Hardware, USA (2016)
23. RITA-BTS: RITA-BTS, Bureau of Transportation Statistics, United States department of Transportation. http://stat-computing.org/dataexpo/2009/the-data.html