Big Data: State-of-the-art Concepts, Techniques, Technologies, Modeling Approaches and Research Challenges

Carlos Costa and Maribel Yasmina Santos

Abstract—**Current advancements in Information Technologies (IT) lead organizations to pursue high business value and competitive advantages through the collection, storage, processing and analysis of huge amounts of heterogonous data, generated at ever increasing rates. Data-driven organizations are often seen as environments wherein the analysis and understanding of products, people and transactions are of major relevance. Big Data, mainly defined as data with high volume, variety and velocity, creating severe limitations in traditional technologies, promises to leverage smarter insights based on challenging and more granular data sources, increasingly demanding emergent skills from data scientists to revolutionize business products, processes and services. The concept gained significant notoriety during the last years, since many business areas can benefit from this phenomenon, such as healthcare, public sector, retail, manufacturing and modern cities. Big Data as a research topic faces innumerous challenges, from the ambiguity and lack of common approaches to the need of significant organizational changes. Therefore, research on Big Data is relevant to assure that organizations have rigorously justified proofs that emergent techniques and technologies can help them making progress in data-driven business contexts. This work presents a state-of-the-art literature review in Big Data, including its current relevance, definition, techniques and technologies, while highlighting several research challenges in this field. Furthermore, this work also provides relevant rules for modelling databases in Big Data environments, which can be used to convert relational data models into column-oriented data models.**

Index Terms—**Big Data, Challenges, Hadoop, Modelling, NoSQL, Review, State-of-the-art, Techniques, Technologies**

I. Introduction

NOWADAYS, we generate data at unprecedented rates, mainly due to the technological advancements we face, namely in cloud computing, internet, mobile devices and embedded sensors [1], [2]. Collecting, storing, processing and analyzing all this data becomes increasingly challenging. Organizations that are able to surpass these challenges and extract business value from Big Data, will gain significant competitive advantages. They will be able to better analyze and understand their products, people and transactions. Big Data is frequently seen as a buzzword for smarter and more insightful data analyses, but one can argue that it is more than that, it is about new challenging and more granular data sources, the use of advanced analytics to create or improve products, processes and services, as well as responding rapidly to business changes [3]. During the last years, there was an increased interest in Big Data [4], and it is sometimes highlighted as fundamental for productivity growth, innovation and customer relationship, benefiting business areas like healthcare, public sector, retail, manufacturing and modern cities, for example [5], [6].

The definition of Big Data is ambiguous, and it is difficult to quantify the level at which data becomes big [7]. Therefore, Big Data is frequently defined by its characteristics (e.g., volume, variety, velocity) and the consequent technological limitations it imposes in organizations, i.e., data is "too big, too fast, or too hard for existing tools to process" [8]. One may argue that if Big Data is data that creates technological limitations, then it always existed and it always will. Currently, a paradigm shift is happening in the way we collect, store, process and analyze data. Organizations need to be aware of these technological trends and strategies that may improve business value. Consequently, Big Data, as a research topic, is of major relevance to assure that organizations have rigorously justified proofs that emergent techniques and technologies can help them making progress in data-driven business environments.

Big Data brings innumerous challenges mainly divided into four categories: general dilemmas, such as the lack of consensus and rigor in the definition, models, architectures or benchmarks, for example; challenges related to the Big Data life cycle, from collection to analysis; security, privacy and monitoring issues; and, finally, organizational change, such as new required skills (e.g., data scientists) or changes in workflows to accommodate the data-driven mindset.

Working with Big Data implies knowledge from multiple disciplines and the term data science is frequently highlighted to designate the area responsible for dealing with Big Data throughout the stages of its life cycle, relying on the scientific method (defining hypothesis and validating conclusions) and on knowledge related to areas like machine learning, programming and databases, for example.

Manuscript received August 10, 2016; revised May 30, 2017. 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 the SusCity project, MITP-TB/CS/0026/2013.

Carlos Costa is with ALGORITMI Research Centre, University of Minho, Campus de Azurém, 4800-058 Guimarães, Portugal (corresponding author, phone: +351-253-510308; fax: +351-253-510300; e-mail: carlos.costa@dsi.uminho.pt).

Maribel Yasmina Santos is with ALGORITMI Research Centre, University of Minho, Campus de Azurém, 4800-058 Guimarães, Portugal (e-mail: maribel@dsi.uminho.pt).

Therefore, in this document, one will refer to data science as the act of extracting patterns and trends from data, through certain data-related techniques, regardless of its characteristics or challenges. These insights can then be communicated or used to create data artifacts or to optimize existing ones, improving business management and performance through data-driven decision-making [9].

This paper presents a state-of-the-art literature review in Big Data, summarizing concepts, techniques, technologies and, more relevant, highlighting current research challenges. Among current literature, there are already some reviews on this topic [6], [10]–[13]. However, the one presented in this paper dedicates significant attention to the definition of Big Data and discusses relevant and innovative Big Data techniques, technologies and research challenges not synthesized in previous literature review papers. This work helps researchers and practitioners in gathering more relevant information for future initiatives. Besides, this work presents a set of rules to model databases in Big Data contexts, converting a relational data model into a column-oriented data model.

This paper is structured as follows: section II aims to define Big Data; section III highlights the main techniques and technologies to design and implement Big Data solutions; section IV describes the automatic procedure to convert relational databases into column-oriented databases for operational Big Data contexts; section V presents Big Data research challenges; section VI concludes with some remarks about the undertaken work and some prospects for future work.

II. What Actually Means Big Data?

First of all, there is no widely accepted threshold for which data becomes big. In [7], the authors attempt to clearly define Big Data by presenting several definitions among the community, highlighting that Big Data is predominantly and "anecdotally" associated with data storage and data analysis, terms dating back to distant times. The authors also argue that the adjective "big" implies significance, complexity and challenge, but also makes difficult to quantitatively define Big Data. The work of [7] presents several definitions, some defining Big Data by its characteristics, others based on the augmentation of traditional data with more unstructured data sources, and others trying to quantify it. They also present definitions that rely on the inadequacy of traditional technologies to deal with this new type of data, presenting several perspectives from the industry, including Gartner, Oracle, Intel, Microsoft and IBM, for example. In order to conclude about the similarity among definitions, the authors state that all definitions include at least one of the following aspects: size; complexity; and techniques/technologies to process large and complex datasets.

The work of [1] attempts to provide a definition: *"Big Data is data that exceeds the processing capacity of conventional database systems. The data is too big, moves too fast, or does not fit the strictures of your database architectures. To gain value from this data, you must choose an alternative way to process it"*. In [6], the authors corroborate this definition by focusing on the fact that traditional software and hardware cannot recognize, collect, manage or process this new type of data in reasonable time. The work of [14] also agrees with these perspectives, defining Big Data by its complexity, speed and several degrees of ambiguity, whose processing is inadequate for traditional methods, algorithms and technologies. Although in [7] the authors are slightly critical both regarding the lack of quantification in the definition of Big Data and the use of data storage and analysis in several attempts to define it, in reality, they conclude by stating that the concept of Big Data includes storage and analysis of large and complex datasets, using a set of novel techniques. The origin of the concept is relatively unknown and its definition evolved rapidly, thus raising uncertainty. In [15], the authors state that size is the characteristic that first stands out, but other characteristics have become usual to define Big Data. In 2001, Doug Laney from Gartner presented the 3Vs model to characterize Big Data by its volume, variety and velocity [16]. IBM and Microsoft based their definitions of Big Data on this model for at least more 10 years [6].

According to [15], volume is a characteristic that indicates the magnitude of data, mentioning that it is frequently reported between Terabytes and Petabytes, citing the survey of [17], in which just over half of the respondents consider datasets bigger than 1TB to be Big Data. However, the authors discuss that data size is relative and varies according to the periodicity and the type of data. It is impractical to define a specific threshold for Big Data volume, since different types of data require different technologies to deal with it (e.g., tabular data and video data), as [15] exemplify. The volume in the 3Vs model characterizes the amount of data that is continuously generated [14], and the main cause for the ever increasing volume is the fact that we currently store all our interactions with the majority of services available in our world [18].

Regarding variety, Big Data can be classified as structured (e.g., transactional data, spreadsheets, relational databases), semi-structured (e.g., web server logs and Extensible Markup Language - XML) and unstructured (e.g., social media posts, audio, video, images) [15], [19]. Traditional technologies can present significant difficulties to store and process Big Data, such as content from web pages, click-stream data, search indexes, social media posts, emails, documents and sensor data. Most of this data does not fit well in traditional databases and there must be a paradigm shift in the way organizations perform analyses to accommodate raw structured, semi-structured and unstructured data, in order to take advantage of the value in Big Data [18].

The final characteristic in the 3Vs model is velocity, referring either to the rate at which data is generated or to the speed of analysis and decision support [15]. Data can be generated at different rates, ranging from batch to real-time (streaming) [18], [19]. It is relevant to apply the definition of velocity to data in motion, instead of applying it to the rate at which data is collected, stored and retrieved from the storage system. Continuous data streams can create competitive advantages in contexts where the identification of trends must occur in short periods of time, as in financial markets, for example [18].

Over time, two additional characteristics emerged: value and veracity. Value represents the expected results of

processing and analyzing Big Data [19], which usually has low value in its raw state, as this is mainly extracted with an adequate analysis [15]. According to [19], value can be obtained through the integration of different data types to improve business and gain competitive advantages. On the other hand, veracity draws attention to possible imprecise data, since sometimes the analysis is based on datasets with several degrees of precision, authenticity and trustworthiness [19]. In [15], the authors corroborate this definition, highlighting the unreliability of certain data sources (e.g., customer sentiments extracted from social media), although recognizing that they can be valuable when adequate techniques and technologies are used.

Other characteristics, not so recognized according to the literature, are the variability and complexity, introduced by SAS [15]. Variability is related to the different rates at which data flows, according to different peaks and inconsistent data velocity. Complexity highlights the challenge of dealing with multiple data sources, namely to connect, match, clean and transform them. Besides these, the work of [14] also proposes three other characteristics: ambiguity, related to the lack of appropriate metadata, resulting from the combination of volume and variety; viscosity, when the volume and velocity of data causes drag in data flows; virality, which measures the time of data propagation among peers in a network. Fig. 1 presents a summary of all these characteristics identified in the literature.

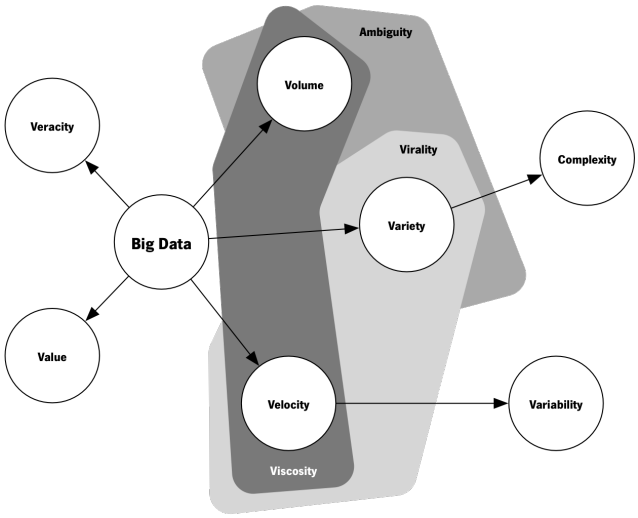

Fig. 1. Main Big Data characteristics identified in the literature.

At this point, it seems that trying to quantify any of these characteristics becomes an impossible task. Big Data remains as an abstract concept [6]. One must accept that it can be a combination of several characteristics, or a strong presence of only one, but it must be recognized as data that make changes in the way we think about techniques and technologies, if they are inadequate to deal with it. It may be a database or a Data Warehouse that cannot scale accordingly on a shared-everything architecture [14], or data mining tasks that cannot be finished without parallel computing. Again, data is *"too big, too fast, or too hard for existing tools to process"* [8]. Defining Big Data by the inadequacy of traditional technologies is relatively dangerous, since advancements are constantly being made (e.g. quantum computers), and such definition implies that Big Data always existed and will continue to exist [7].

The current definitions of Big Data are relatively dependent on the techniques and technologies to collect, store, process and analyze it. These will evolve over time and we need to learn to live with it. It will always be a matter of analyzing new technological trends that may benefit business and reconsider new strategies related to data. Currently, a new paradigm shift is happening. It does not need to be a change in all organizations, but scientific progress related to Big Data will continue to exist, in order to assure that organizations have rigorously justified proofs that emerging techniques and technologies can help them making progress in data-driven business.

III. Techniques and Technologies for Big Data

This section presents techniques and technologies identified in the literature, which are adequate to support the design and implementation of Big Data solutions.

A. Designing Big Data Solutions

According to [10], citing [20] and [21], a Big Data solution generally contemplates the following principles: present high level architectures, addressing the distinct role of specific technologies; include a variety of data science tasks, such as data mining, statistical analysis, machine learning, real-time visualization, and in-memory analysis; combine the benefits of different tools for different tasks; do not move data, bring analysis closer to data; distribute processing and storage across different nodes in a cluster; and assure coordination between data and processing nodes to improve scalability, efficiency and fault-tolerance. There are several considerations throughout the life cycle of Big Data, significantly different from traditional environments. Dealing with Big Data requires new approaches, which are discussed in this subsection.

1) The Big Data Life Cycle

According to a survey including several analysts at Microsoft [22], Big Data analytics tasks can be grouped into five steps: acquire data; choose the architecture based on cost and performance; shape the data according to the architecture; write and edit code; reflect and iterate on the results. Processing Big Data for analysis typically differs from processing traditional transactional data. As [14] claims, in traditional environments, data is explored, a model is designed and a database structure is created. However, in Big Data environments, data is first collected and loaded to a certain storage system, a metadata layer is applied and then a structure is created. There is no need to start by transforming data to properly fit a relational model, as transformations only occur after having everything stored in efficient storage systems. This represents a departure from a traditional Extraction, Transformation and Loading (ETL) approach to an Extraction, Loading and Transformation (ELT) approach. Fig. 2 presents the Big Data processing flow according to [14].

The Big Data processing flow starts by gathering data from multiple sources, such as Online Transaction Processing (OLTP) systems, multiple files, sensors and the Web. This data is then stored in a landing zone capable of handling the volume, variety and velocity of data, which is

typically a distributed file system. Data transformations must occur on data stored in the landing zone, fulfilling the requirements of efficiency and scalability, and the subsequent results can then be integrated into analytical tasks, operational reporting, databases, or raw data extracts. In this context, [23] also mention relevant best practices regarding the Big Data life cycle:

- Plan a "data highway" with multiple caches - raw source (immediate), real-time cache (seconds), business activity cache (minutes), top line cache (24 hours) and data warehouse or long time series cache (daily, periodic and annual). Data will flow through these different caches, according to the business needs;
- Use Big Data analytics to enrich data before moving it to the next cache. For example, produce numeric sentiments from mining unstructured tweets. The opposite is also true, so that earlier caches can benefit from the less granular ones. In [23], the authors claim that the performance implications of this enrichment should be further evaluated, since data should be moved from the raw source to the real-time cache according to the established time thresholds. Also, we can store multiple data sources, make them available for querying, manipulate them, use them to serve business and then archive them;
- Adjust the data quality needs according to the latency requirements, i.e., complex data quality jobs take more time to complete than simpler ones focusing individual values. However, [23] also suggest that one should add value to data as soon as possible, using data integration tasks and including results from data mining, for example. There must be a balance between latency and business value;
- Big Data streaming analytics can be relevant to certain data flows, analyzing data and taking actions as it flows through continuous data streams [24]. In-database analytics can also be a relevant capability to exploit, as [23] highlight.

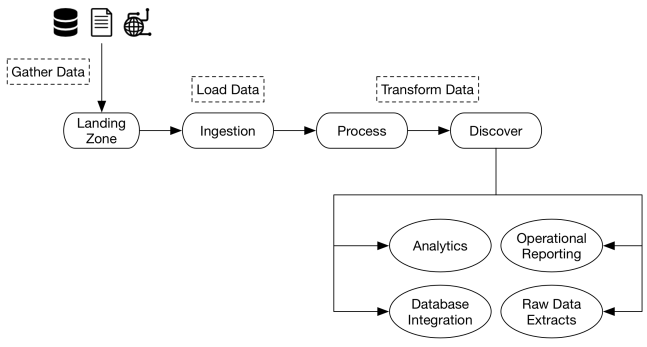

Fig. 2. Big Data processing flow. Redrawn based on [14].

In [25], these perspectives are complemented, since the authors state that several analytical mechanisms should be included in Big Data solutions, ranging from statistical analysis to data mining and visualization. Moreover, processed data and new insights can be made available using open and recognized standards, interfaces and web services. Regarding Big Data analytics, there is a vast set of available techniques that can be used to extract value from data. Data mining techniques, such as clustering, association rules, classification and regression are still present in Big Data environments [5], now with the challenge of distributing them to perform at scale [10], [13]. Achieving scalability in these techniques is what makes Big Data analytics different from traditional data analytics. The range of analytical mechanisms and the ambiguous terms to define them may lead to a completely new buzzword: Data Science. Techniques such as sentiment analysis, time series analysis/forecasting, spatial analysis, optimization, visualization or unstructured analytics (e.g., text, audio, video) [15], can all be present in the knowledge base of a Data Scientist [9]. These techniques are relevant in the Big Data life cycle to extract value from it.

2) Architectural and Infrastructural Requirements

The different steps to process Big Data, presented above, must be performed in Big Data environments, according to several requirements identified by [14]:

- Absence of fixed data models, to adequately accommodate the complexity and size of data, regardless of its characteristics;
- Scalable and performant systems to collect and process data either in real-time or in batches;
- The architecture should support data partitioning due to the volume of data;
- Data transformations use scalable, efficient and fault-tolerant mechanisms. The results should be stored in adequate systems, such as distributed file systems or non-relational database systems. Data reads should be efficient;
- Data should be replicated and shared across multiple nodes, to support fault-tolerance, multistep processing and multipartitioning.

The work of [23] corroborates most of the requirements from [14], and add the following capabilities expected from Big Data environments: possibility to implement User-Defined Functions (UDFs) in several programming languages and to execute them over huge datasets within minutes; load and integrate data at high rates; execute queries on streaming data; schedule tasks on large clusters; and support mixed workloads, including several ad hoc queries or strategic analysis from multiple users, while loading data in batches or in a streaming fashion.

Big Data solutions should be supported by an adequate infrastructure. Regarding this requirement, organizations can currently rely on cloud computing, either by using private, public or hybrid clouds [26], in order to provide the underlying resources for massive computations [27]. Cloud models, such as Infrastructure as a Service (IaaS), become relevant to accomplish several requirements in Big Data infrastructures, including scalability, commodity hardware, elasticity, fault-tolerance, self-manageability, high throughput, fast I/O and a high degree of parallelism [14], [28]. Commodity hardware plays a relevant role in Big Data infrastructures, namely due to the lower costs in building shared-nothing architectures. Google's own papers about the Google File System (GFS) [29], MapReduce [30] and Bigtable [31], served as inspiration for most of these requirements and for several Big Data technologies that will be presented later.

In [23], the authors argue that a traditional Relational Database Management System (RDBMS) is not suitable for a wide range of Big Data use cases, due to the requirements identified above (e.g., search ranking, sensors, social customer relationship management, document similarity testing, loan risk analysis). The work of [14] also claims that DWs based on traditional RDBMSs have several design limitations that imply architectural and infrastructural changes to process Big Data, since they cannot be distributed as efficiently as non-relational systems due to Atomicity, Consistency, Isolation, Durability (ACID) compliance rules and due to the fact that data partitioning in these systems often does not necessarily mean more scalability or workload reduction. Furthermore, the author mentions the fact that in many of these systems the processor and memory are often underused, and the way queries are designed typically increases the workload, such as executing star schema queries on a third normal form database model, generating significant volume of I/O and inadequate network throughput.

The work of [23] presents the capabilities that existing RDBMSs vendors are including to extend their solutions for Big Data environments. The authors compare these extended versions with the most commonly recognized open source implementation of MapReduce, namely Apache Hadoop. The authors highlight that Hadoop is open source, less expensive, has a more flexible storage (unstructured data), is adequate for massive scans and has deep support for complex structures. However, the authors claim that Hadoop only has indirect support for relational semantics and little or no support for transaction processing, when compared to these extended RDBMSs.

3) Common Architectures for Big Data Solutions

There are two Big Data architectures highlighted in the literature, namely the Lambda Architecture (Fig. 3) and the National Institute of Standards and Technology (NIST) Big Data Reference Architecture (Fig. 4).

The main idea behind the Lambda Architecture [20] is to think of a Big Data system as a series of layers that satisfy particular needs. The architecture is divided into three main components: batch, serving and speed layers. In the batch layer, a master dataset stores all the data. Since it is unthinkable to read a dataset with possible petabytes of data every time a query is executed, the architecture contains batch views in the serving layer, which are precomputations of the master dataset. Instead of executing the query every time and scan the entire master dataset, the results are returned from batch views with indexing support, thus random reads are possible. Therefore, the batch layer is not only responsible for storing an immutable, constantly growing master dataset, but also for computing functions on the same. However, with only these two layers, batch views would be quickly outdated, since new data takes time to propagate from the batch layer into the serving layer. This does not meet the requirements of low latency (real-time) environments. Consequently, the authors propose the speed layer, which aims to compute functions on data in real-time.

The NIST Big Data Public Working Group, namely the Reference Architecture Subgroup, has been working on an open reference architecture for Big Data [32], in order to create a tool to facilitate the discussion of requirements, design structures and operations inherent in Big Data environments. According to the authors, the NIST Big Data Reference Architecture is not a system architecture, but rather a common reference, which is not coupled with specific vendors, services, implementations or any specific solutions. The architecture includes several components, as can be seen in Fig. 4, each with specific tasks in a Big Data solution, being extensively described in [32].

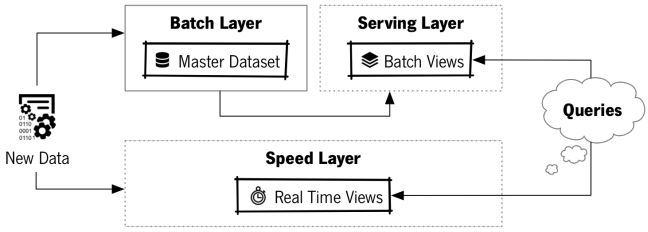

Fig. 3. The Lambda Architecture. Redrawn based on [20].

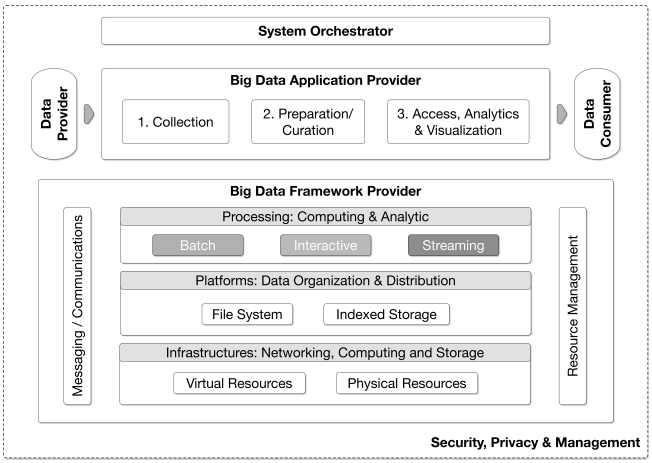

Fig. 4. The NIST Big Data Reference Architecture. Redrawn based on [32].

B. Hadoop and Related Projects

As already mentioned, Hadoop is an open source Apache project based on GFS and MapReduce [33]. Hadoop contains two main components: the Hadoop Distributed File System (HDFS) and a distributed processing framework named Hadoop MapReduce. Hadoop can store and process vast amounts of data by distributing storage and processing across a scalable cluster of multiple nodes built with commodity hardware. In HDFS, files are divided into blocks distributed and replicated across nodes. HDFS assures many requirements identified above, such as fault-tolerance and availability, for example. Hadoop MapReduce is a programming model and an execution engine for processing large datasets stored in HDFS, based on the divide and conquer method, thus dividing a complex problem into many simpler problems and then combining each simpler solution into an overall solution to the main problem. These are called the Map and Reduce steps [10]. Regarding HDFS, there are two types of nodes in the cluster: a NameNode, which is responsible for storing metadata about blocks and nodes; and a DataNode, which stores data blocks [33]. Regarding Hadoop MapReduce, there are also two types of nodes, a JobTracker that schedules jobs and distributes tasks across slaves called TaskTrackers [10].

Over the years Hadoop has evolved considerably, including the transition from MapReduce to YARN (or

MapReduce 2.0) [27]. YARN rethinks the JobTracker and TaskTracker components, replacing them with a ResourceManager, a NodeManager and an ApplicationMaster, to solve some problems in Hadoop MapReduce, such as scalability on large clusters or support for alternative programming paradigms [14]. Apart from that, Hadoop has several related projects, as Fig. 5 demonstrates, highlighting their main features [34].

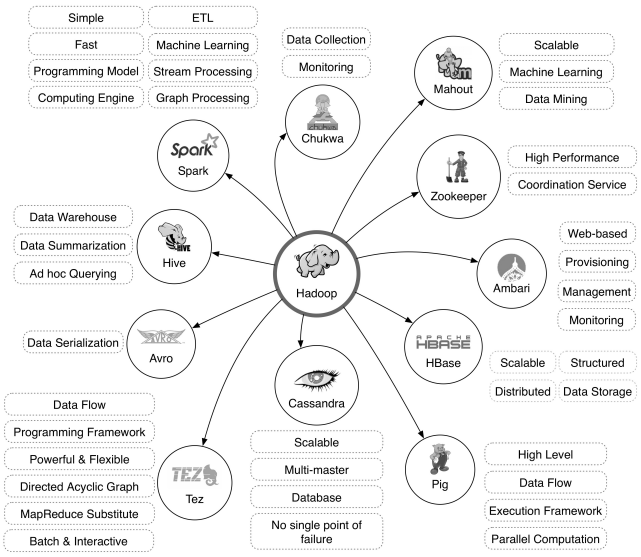

Fig. 5. The rich ecosystem of Apache Hadoop.

Other related projects not present in Fig. 5 may include: Flume, a service to collect, aggregate and move large amounts of data; Oozie, a workflow and coordination system for jobs in Hadoop; HCatalog, a metadata layer for data stored in Hadoop, built on top of the Hive metastore; Sqoop, a connector to integrate data from other existing platforms, such as the Data Warehouse, metadata engines, enterprise systems and transactional systems [14]. There are also more projects that can interact with Hadoop's interfaces or be co-located with it, such as projects for real-time stream processing or interactive ad hoc analysis. Since real-time data processing is becoming increasingly relevant to organizations [19], Storm is a real-time computation system to process streams with high throughput and low latency. Kafka, on the other hand, is a messaging/queuing system to send and consume messages between processes, in an asynchronous manner and with fault-tolerance [20]. Interactive and low latency ad hoc analysis over large datasets is also a relevant scenario in organizations. Occasionally, users do not know queries in advance and need to execute ad hoc queries within seconds, even at scale. Apache Drill is a distributed system for interactive ad hoc analysis of large datasets. It is a query mechanism that can interact with several data sources, including HDFS and distributed databases [19].

Still related to Hadoop, there are several security projects. In [35], five pillars for security in Hadoop are established: administration, authentication, authorization, audit and data protection. Kerberos, Apache Knox and Apache Ranger are highlighted as projects related to these five pillars, in order to assure a secure Hadoop environment. Kerberos can be used to authenticate users and resources within Hadoop clusters. Apache Knox complements Kerberos, by blocking services at the perimeter of the cluster and hiding the cluster's access points from end users, thus adding another layer of protection for perimeter security. Finally, Ranger provides a centralized platform for policy administration, authorization, audit and data protection (e.g., encrypted files in HDFS).

C. Distributed Databases

Database technology has evolved significantly towards handling datasets at different scales and supporting several applications that may have high needs for random access to data [6], [27]. Not only SQL (NoSQL) databases have become popular mainly due to the lack of scalability in RDBMSs, since this new type of databases provides mechanisms to store and retrieve large volumes of distributed data [27]. The relevant factors that motivated the appearance of NoSQL databases were the strictness of the relational model and the consequent inadequacy to store Big Data. NoSQL databases are seen as distributed, scalable, elastic and fault-tolerant storage systems. They satisfy an application's need for high availability even when nodes fail, appropriately replicating data across multiple machines [24]. Relational databases will certainly evolve and some organizations (e.g., Facebook) are using mixed database architectures [6]. Combining the benefits of both storage systems is a current research trend [28]. A recent term is emerging, NewSQL, which combines the relational data model with the benefits of NoSQL systems, such as scalability [36]. According to [37], NoSQL and NewSQL databases are mainly designed to scale OLTP-style workloads over several nodes, fulfilling the requirements of environments with millions of simple operations (e.g., key lookups, reads/writes of one record or a small number of records).

This phenomenon changed the way databases are currently designed. While a RDBMS complies to ACID properties [14], a NoSQL database, as a distributed system, typically follows the considerations of the Consistency, Availability, Partition tolerance (CAP) theorem: "any networked shared-data system can have at most two of three desirable properties" [38]. These properties include: consistency, equivalent to a single up-to-date copy of the data; high availability of that data; and tolerance to network partitions. As [38] claims, CAP served its purpose to leverage the design of a wider range of systems and tradeoffs, in which the NoSQL movement is a clear example. The fact that one has to choose 2 of the 3 properties was always misleading, sates the author, since it tends to simplify the "tensions among properties". These properties are more continuous than binary and therefore they can have many levels. CAP only prohibits perfect availability and perfect consistency in the presence of network partitions.

There are several NoSQL databases, so enumerating and evaluating all of them becomes a nearly impossible task. In [39], it is stated that over 120 NoSQL databases were known in 2011. Taking this into consideration, NoSQL databases are typically divided into four data models, which are described as follows along with several examples:

- Key-value model - values are typically stored in key-value pairs. The key uniquely identifies a value of an arbitrary type. These data models are known for being schema-free, but may lack the capability to adequately represent relations or structures, since queries and indexing are assured through the key [36]. Every key is unique and queries are tightly coupled with keys [6]. Examples: Redis; Memcached; BerkeleyDB; Voldemort; Riak; Dynamo;
- Column-oriented model - a columnar data model can be seen as an extension of the key-value model, adding columns and column families, and providing more powerful indexing and querying due to this addition [14]. This design was largely inspired by Bigtable [6], [36], which does not mean that all column-oriented databases are fully inspired by it (e.g., Cassandra adopts design aspects from both Dynamo and Bigtable). Examples: Bigtable; HBase; Cassandra; Hypertable.
- Document model - suitable for representing data in document format. JavaScript Object Notation (JSON) is here frequently used. It can contain complex structures, such as nested objects, and also typically includes secondary indexes, thus providing more query flexibility than the key-value data model [36]. Examples: MongoDB; CouchDB; Couchbase.
- Graph model - based on the graph theory, in which objects can be represented as nodes, and relationships between them represented as edges [14]. Graphs are specialized in handling interconnected data with several relationships [36]. Examples: Neo4j; InfiniteGraph; GraphDB; AllegroGraph; HyperGraphDB.

Regarding NewSQL, as the name implies, these databases are based on the relational model [36], offering either a pure relational view of the data (e.g., VoltDB, Clustrix, NuoDB, MySQL Cluster, ScaleBase, ScaleDB) or similar (e.g., Google Spanner). According to [36], sometimes, interactions with these databases occur in terms of tables and relations, but they might use different internal representations, which is the case for NuoDB. Different NewSQL databases support different SQL compatibility, such as unsupported clauses or other incompatibilities with the standard. Similar to NoSQL, NewSQL databases can scale accordingly by adding nodes to the cluster.

D. Other Technologies for Big Data Analytics

By describing Hadoop and its related projects, several technologies for Big Data analytics were already inherently identified: streaming analytics (e.g., Spark Streaming and Storm); data mining and machine learning (e.g., Spark MLlib and Mahout); Data Warehousing (e.g., Hive); interactive ad hoc analysis (e.g., Drill); data flow (e.g., Pig). However, no data visualization tools were presented yet.

Regarding Big Data visualization, several mashup tools can be highlighted, such as Datameer, FICO Big Data Analyzer (former Karmasphere), Tableau and TIBCO Spotfire [14]. These mashup tools can integrate data from multiple sources into a single picture. As [14] highlights, there is also the possibility to visualize Big Data with statistical tools, like R or SAS, for example, taking advantage of their other capabilities. Other tools are also briefly mentioned in the literature, such as Jaspersoft Business Intelligence (BI) Suite and Pentaho Business Analytics [10]. Certainly, many other visualization tools exist and may be adequate for Big Data visualization, such as Excel and its Power View extension [40], JavaScript libraries or Python's plot capabilities [41].

Besides data visualization, there are other tools to extract, load, transform and integrate data before analytical tasks. An example of such tool is Talend Open Studio for Big Data [10]. Apart from the afore mentioned tools related to Hadoop for data mining and machine learning, other alternatives identified in the literature may include: MADLib and EMC Greenplum [25]; R, MOA, WEKA and Vowpal Wabbit [13]; data mining tools from SAS or IBM [14]; Rapidminer; and KNIME [6]. Some of these tools, like R and WEKA, for example, are not scalable by default and they are also used in traditional data mining and machine learning environments, where processing large training sets is not a significant concern. Over time, these tools were extended with several connectors for scalable Big Data stores and packages for distributed processing (e.g., SparkR, RHadoop, RHive, distributedWekaHadoop), but by default, without these extensions, they are better suited for small to moderate datasets. This does not mean that they are not useful in Big Data mining, quite the opposite, but the volume of data that serves as training and testing sets should be considered (preprocessing large datasets can be useful in these cases). The same principle applies to other algorithms implemented in any other language like Python or Java, for example. It must be remembered that one of main challenges regarding the Big Data life cycle is to scale the algorithms to extract value from data [27].

IV. Data Modeling

As already highlighted in this paper, in a Big Data context, the ability to collect, process and store data increases with the use of NoSQL databases. These databases are characterized by being schema-free, allowing the storage of huge amounts of data without many concerns about its structure. These concerns usually emerge later on a schema-on-read approach, in which data is parsed, formatted and cleaned at runtime. Although having fewer concerns at the beginning of the collection phase, this adds several tasks latter when there is the need to develop specific data management applications. At some point, these schema-free repositories need to be transformed into some structured data model that allows data manipulation and analysis by the users. Our previous work described in [42] introduced some modeling concepts in a Big Data environment, proposing a set of rules for the automatic transformation of a relational data model into an HBase columnar format and, also, proposing transformation rules for identifying Hive tables.

Although in [42] the identification of several data structures suited to be implemented in HBase was achieved, not all the descriptive tables needed to support organizations in their day-by-day activities were identified. In this paper, we enhance and extend the set of rules needed to transform

an operational data model, based in a relational schema, into an HBase model, identifying all the needed descriptive and analytical tables. The descriptive tables give support to operational activities, while the analytical tables give support to operational and analytical activities in the organization, namely supporting the decision-making process. These analytical tables are intended to answer queries that manipulate hundreds or thousands of records, and not millions or billions, as in this case the implementation of a Big Data Warehouse in Hive would be more appropriate [43].

For the transformation of a relational data model into a columnar data model, namely for HBase, it is necessary to consider that a column-oriented database is constituted by a set of tables integrating rows, but organized by groups of columns usually named column-families. This organization makes a vertical partitioning of the data. Each column-family may contain a variable number of columns and allows the lack of some columns between different rows of the same table [44].

Given this context, the following definitions formalize the concept of a relational data model and a columnar data model.

Definition 1. A Relational Data Model, $RDM = (T, A)$, includes a set of tables $T = \{T^1, T^2, \dots, T^n\}$ and the corresponding attributes, $A = \{A^1, A^2, \dots, A^n\}$, where A^1 is the set of attributes for table T^1, $A^1 = \{A_1^1, A_2^1, \dots, A_k^1\}$. Tables in a data schema are linked to each other through relationships, being cardinalities of type $1{:}\,n$, $m{:}\,n$ or $1{:}\,1$, with the optional 0 when needed. Each table includes an attribute that represents the primary key of the table (PK) and may include one or more attributes representing foreign keys (FK), linking this table to other tables in the data schema.

Definition 2. A Columnar Data Model, $CDM = (T, CF)$, includes a set of tables $T = \{T^1, T^2, \dots, T^n\}$. Each table integrates a key and a set of column-families, as $T^i = \{key^i, CF^1, \dots, CF^m\}$. Each column-family integrates a set of columns representing the atomic values to be stored, $CF^j = \{C_j^1, \dots, C_j^k\}$.

Considering these definitions, the following rules can be adopted for the transformation of a relational data model into a columnar data model. This set of rules can also be used to make the necessary transformations for other NoSQL databases, based in the columnar format, like Cassandra, as long as the necessary adaptations are performed, as the organization of the columns, or the way the key attribute of each table is defined, may be slightly different.

Rule CDM.1. Identification of Column-Families. The identification of column-families of a *CDM* follows a two-step approach.

Rule CDM.1.1. Identification of Descriptive Column-Families. All tables that do not include any FK in the *RDM* correspond to descriptive column-families in the *CDM*, as these tables are core tables storing the data associated with the main entities of the application domain, like *Students*, *Teachers*, *Facilities*, among others. These are the core descriptive column-families that may also help identifying the complementary descriptive column-families. These complementary column-families are derived from tables that are linked to core tables through a $1{:}\,n$ relationship, from the core table to the complementary table, and are tables usually split in the normalization process. Usually, these tables are used to complement the description of the core entities. As an example, in the relationship *a Department has many Teachers*, *Department* is the core table and *Teachers* the complementary table. These complementary tables are quite static over time, meaning that the set of *Teachers* of a *Department* is not changing daily, for instance. This represents a type of relationship different from *a Student has many Evaluations*, for example, as with time a *Student* will be enrolled in many evaluations. Rule CDM.1.1 does not handle this last type of relationship, in which the entities are used to feed business processes and not to describe the main entities of the application domain. After the identification of the descriptive column-families, both core and complementary, their columns must also be identified. The columns of a descriptive column-family are constituted by the set of non-key attributes (excluding primary or foreign keys) of the corresponding tables in the *RDM*.

Rule CDM.1.2. Identification of Analytical Column-Families. All tables present in the *RDM* not identified by Rule CDM.1.1 as descriptive column-families give origin to analytical column-families, integrating the day-by-day activities of an application domain or the main changes in the characterization of the core entities of the application domain. These analytical column-families will be stored, processed and analyzed considering the several descriptive column-families. Rule CDM.1.2 excludes tables from the *RDM* that do not include any attributes beside keys (primary or foreign). The columns of analytical column-families are constituted by the set of non-key attributes (excluding primary or foreign keys) of the corresponding tables in the *RDM*.

Rule CDM.2. Identification of Tables. Two types of tables are proposed for a *CDM*, descriptive and analytical tables.

Rule CDM.2.1. Descriptive Tables. Descriptive tables are those tables that support specific data management tasks in an operational system. Each descriptive column-family identified by Rule CDM.1.1 gives origin to a descriptive table.

Rule CDM.2.2. Analytical Tables. For the identification of the set of analytical tables there is the need of identifying the data workflows present in the *RDM*. For identifying the data workflows of a *RDM*, all tables present in the *RDM* not identified by Rule CDM.1.1 as descriptive column-families start a data workflow following the *n:1* relationships associated to them, and all other *n:1* relationships that follow. A data workflow ends when no other *n:1* relationships are found, meaning that it was possible to join a coherent piece of information that is related with each other and that was split in the *RDM* by the normalization process. This means that a specific table in the *RDM* starts a data workflow and two or more tables without any FK end the data workflow. All identified workflows give origin to analytical tables.

Rule CDM.3. Integration of Column-Families into Tables. A specific table integrates a key, a very important component of a table in a *CDM*, and a set of column-families, which may vary depending on the table's type.

Rule CDM.3.1. Column-Families of Descriptive Tables. A descriptive table derived from a main descriptive column-family will include this main descriptive column-family as its unique column-family. A descriptive table derived from a complementary descriptive column-family will include as column-families the complementary and the core descriptive column-families to which it is associated. In case of multiple *1:n* relationships between, for instance, many main descriptive column-families and a complementary descriptive column-family, the related set of column-families is integrated in the descriptive table.

Rule CDM.3.2. Column-Families of Analytical Tables. An analytical table includes as column-families the descriptive column-families and, if applicable, the analytical column-families associated with the tables of the *RDM* included in the data workflow that gave origin to a specific analytical table by Rule CDM.2.2.

Rule CDM.4. Definition of the Tables' Key. A table's key should be able to assure an adequate performance throughout read and write access patterns from the applications. The key represents a set of one or more attributes (concatenated) that has the potential to form a natural key that properly identifies each row in the *CDM*. This key must serve the applications' get, scan and put patterns, keeping them as short as possible, while maintaining the potential for adequate access patterns [44]. The order in which the attributes are concatenated plays a relevant role in the design of the key, since HBase stores keys in a sorted order [44].

For demonstrating the usefulness of the proposed set of rules, let us take as an example the data model used in [42] and see how the rules here proposed advance the previous ones, deriving a set of HBase tables that completely complement each other and provide a coherent support for business applications.

The *RDM* used as starting point is presented in Fig. 6. The model integrates eight tables, several attributes, and the relationships among the tables, represented by the corresponding foreign keys.

Starting by Rule CDM.1, in particular Rule CDM.1.1, two types of descriptive column-families are identified. The core descriptive column-families are $Hotels_{CF}$, $POIs_{CF}$, $Guests_{CF}$ and $Amenities_{CF}$, as these tables only receive relationships with cardinality of 1, not integrating any FK. One complementary descriptive column-family is identified, associated with the *Rooms* table, giving origin to the $Rooms_{CF}$. A table in the model gives origin to an analytical column-family (Rule CDM.1.2), namely $Reservations_{CF}$. In this model, *HotelPOIs* and *RoomAmenities* are not identified as analytical column-families as they do not include any other attribute besides keys. The attributes of the identified column-families are the attributes of the corresponding tables in the *RDM*, excluding the keys, either PK or FK.

Regarding Rule CDM.2, the descriptive tables are $Hotels_T$, $Rooms_T$, $POIs_T$, $Guests_T$ and $Amenities_T$ associated with the identified descriptive column-families (Rule CDM.2.1), while for the identification of the analytical tables there is the need of identifying the data workflows of the *RDM* (Rule CDM.2.2).

Using all the tables present in the *RDM* that were not identified as descriptive column-families, *HotelPOIs*, *RoomAmenities* and *Reservations*, three data workflows are identified (Fig. 7) following the *n:1* relationships starting in these tables and following this type of relationship until no others *n:1* relationships are found. Following this procedure, the three workflows originate three analytical tables, from now on named as $Reservations_T$, $RoomAmenities_T$ and $HotelPOIs_T$.

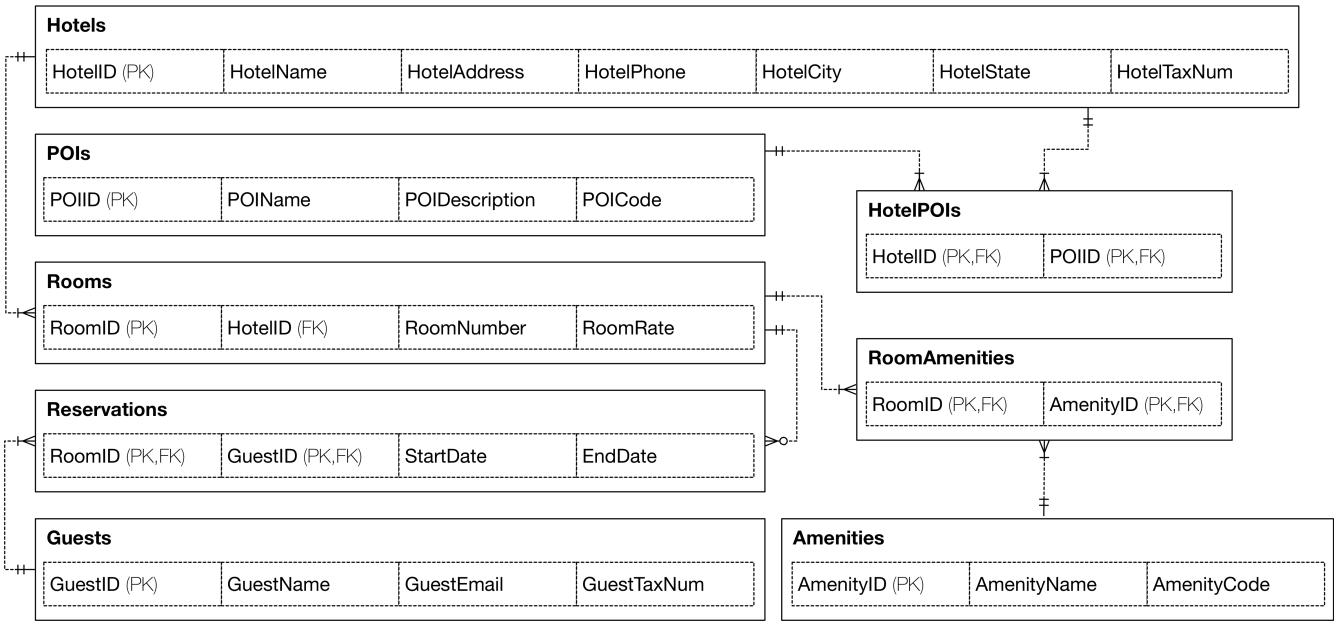

Fig. 6. *RDM* for the Hotel's Demonstration Case. Redrawn based on [45], p. 63.

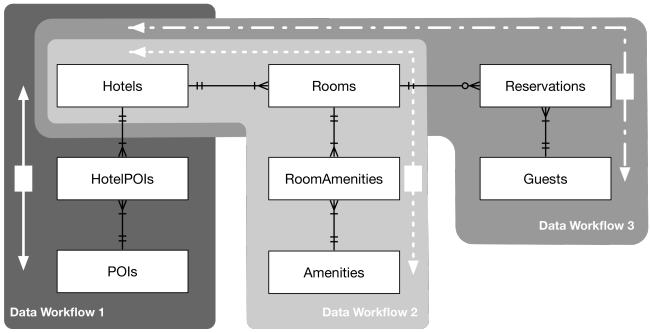

Fig. 7. Data Workflows for the Hotel's Demonstration Case.

Having identified the column-families and the several tables, it is now necessary to assign to each table its respective column-families (Rule CDM.3.1). Each descriptive table integrates the core column-family, or a set of column-families including the core and the complementary descriptive column-families, from which it was derived. Taking $Rooms_T$ as an example, as this table was derived from the complementary descriptive column-family $Rooms_{CF}$, besides this column-family, the HBase table also needs to include the column-family associated with the $Hotels_{CF}$, which is the core descriptive column-family. $Hotels_T$, $POIs_T$, $Guests_T$ and $Amenities_T$ include the corresponding core column-families, namely $Hotels_{CF}$, $POIs_{CF}$, $Guests_{CF}$ and $Amenities_{CF}$.

For the analytical tables (Rule CDM.3.2), $Reservations_T$ integrates four column-families, $Hotels_{CF}$, $Rooms_{CF}$, $Reservations_{CF}$ and $Guests_{CF}$; $RoomAmenities_T$ integrates three column-families, $Hotels_{CF}$, $Rooms_{CF}$ and $Amenities_{CF}$; and, $HotelPOIs_T$ integrates two column-families, $Hotels_{CF}$ and $POIs_{CF}$.

After following the proposed rules, the identified columnar data model for HBase includes five descriptive tables and three analytical data tables, as depicted in Fig. 8 showing the proposed keys (Rule CDM.4) and columns.

The obtained data model allows data management for all descriptive tables in terms of inserting/updating/deleting hotels, rooms, guests, amenities and POIs, as well as upgrading new facts generated by the organization's day-by-day activities, namely new reservations, new amenities for the available rooms or new POIs characterizing the hotels' surroundings. Besides adding these new facts, it is possible to query the available data, answering questions that the operational or tactical managers may have, supporting their decision-making processes.

Looking to the obtained model and comparing it with the one presented in [45], it is possible to verify that although both models present a similar number of tables (seven vs. eight), they are organized in a different way. While the model presented here includes descriptive tables for inserting new POIs or amenities, the model proposed in [45] only includes base tables for hotels, guests and rooms. Reservations and POIs by hotel are organized in a similar way in both models.

The main difference is associated with the possibility of managing all the available data using the approach proposed here, against the possibility of having some specific tables for querying the data, like the available rooms or hotels by city, using the model proposed in [45]. Nevertheless, both data models are able to answer the same questions, as they include the same base columns.

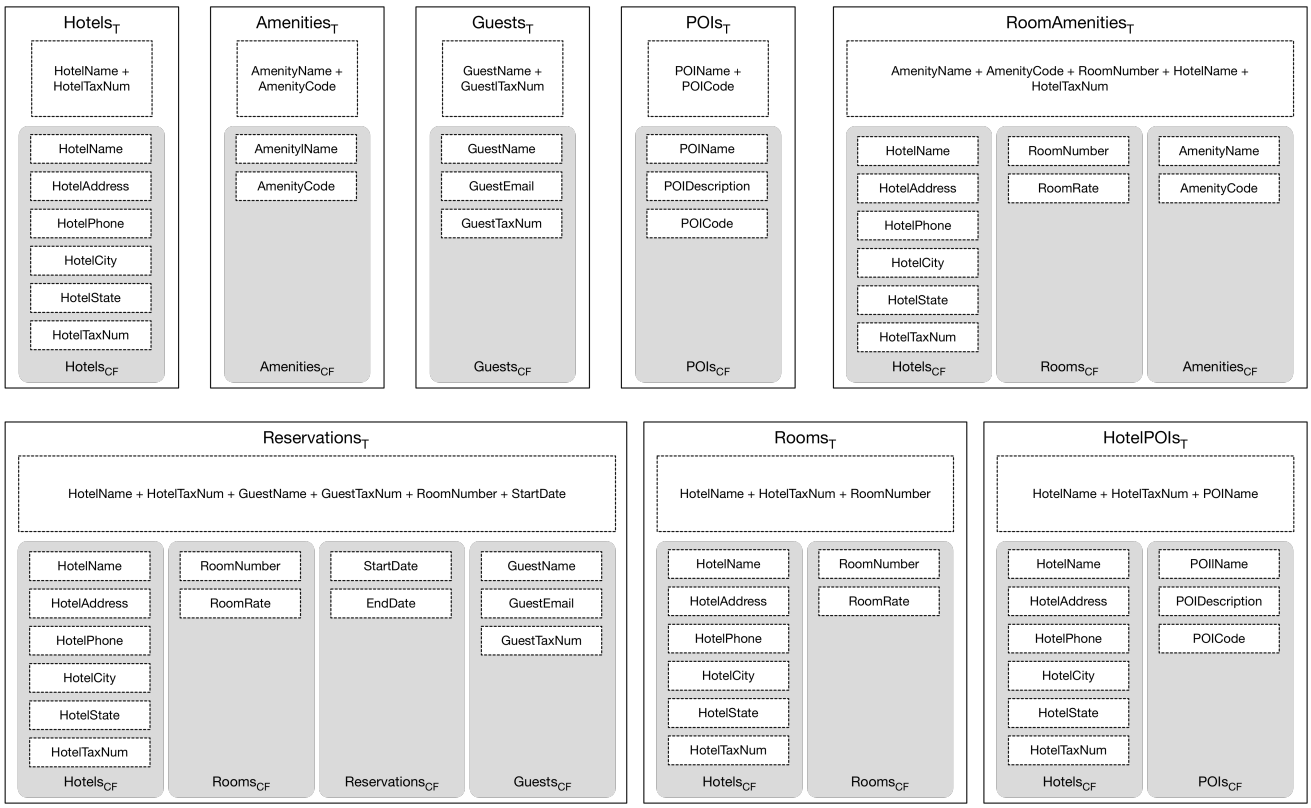

Fig. 8. *CDM* for the Hotel's Demonstration Case.

At this point, it is important to mention that while the data model here obtained (Fig. 8) is suited for HBase, the one proposed in [45] is suited for Cassandra, highlighting the many different ways of organizing a columnar data schema.

After showing how the proposed rules can be applied, a similar exercise is now followed for transforming a well-known *RDM* into a *CDM*. The TPC Benchmark™ H (TPC-H) is a decision support benchmark that includes a set of business oriented ad-hoc queries and concurrent data modifications [46]. It is usually used for benchmarking decision support systems that process large volumes of data, justifying the transformation of its data model into a *CDM*. The obtained *CDM* can afterwards be used for benchmarking processing technologies with data stored in denormalized tables.

The data model of the TPC-H benchmark is depicted in Fig. 9. As can be seen, this data model includes eight transactional tables dealing with sales, customers and suppliers, modelling a business that manages, sells and distributes products.

Following the proposed rules, and starting by Rule CDM.1, Rule CDM.1.1, two types of descriptive column-families are identified. The core descriptive column-families are $Region_{CF}$ and $Part_{CF}$, as these tables only receive relationships with cardinality of 1, not integrating any FK. Three complementary descriptive column-families are identified, giving origin to $Nation_{CF}$, $Supplier_{CF}$ and $Customer_{CF}$. Three other tables give origin to analytical column-families (Rule CDM.1.2), namely $PartSupp_{CF}$, $Orders_{CF}$ and $LineItem_{CF}$. The attributes of the identified column-families are the attributes of the corresponding tables in the *RDM*, excluding the keys, either PK or FK.

Regarding Rule CDM.2, the descriptive tables are $Region_T$, $Part_T$, $Nation_T$, $Supplier_T$ and $Customer_T$ associated with the identified descriptive column-families (Rule CDM.2.1), while for the identification of the analytical tables there is the need of identifying the data workflows of the *RDM* (Rule CDM.2.2).

Using all the tables present in the *RDM* that were not identified as descriptive column-families, *PartSupp*, *Orders* and *LineItem*, three data workflows are identified (Fig. 10) using the *n:1* relationships starting in these tables and following this type of relationship until no more *n:1* relationships are found. Following this procedure, the three data workflows originate three analytical tables, named as $PartSupp_T$, $Orders_T$ and $LineItem_T$.

Having identified the column-families and the several tables, it is now necessary to assign to each table its respective column-families. Each descriptive table (Rule CDM.3.1) integrates a core column-family, or a set of column-families including the core and the complementary descriptive column-families, from which it was derived.

Taking $Customer_T$ as an example, and as this table was derived from the complementary descriptive column-family $Customer_{CF}$, besides this column-family, the HBase table also needs to include the column-families associated with it, which are the $Nation_{CF}$ complementary descriptive column-family and the $Region_{CF}$ core descriptive column-family. $Region_T$ and $Part_T$ include the corresponding core column-families, $Region_{CF}$ and $Part_{CF}$, *respectively*. $Nation_T$ includes the core $Region_{CF}$ and the complementary $Nation_{CF}$. $Supplier_T$ includes $Nation_{CF}$ and $Supplier_{CF}$ as complementary column-families and $Region_{CF}$ as core column-family.

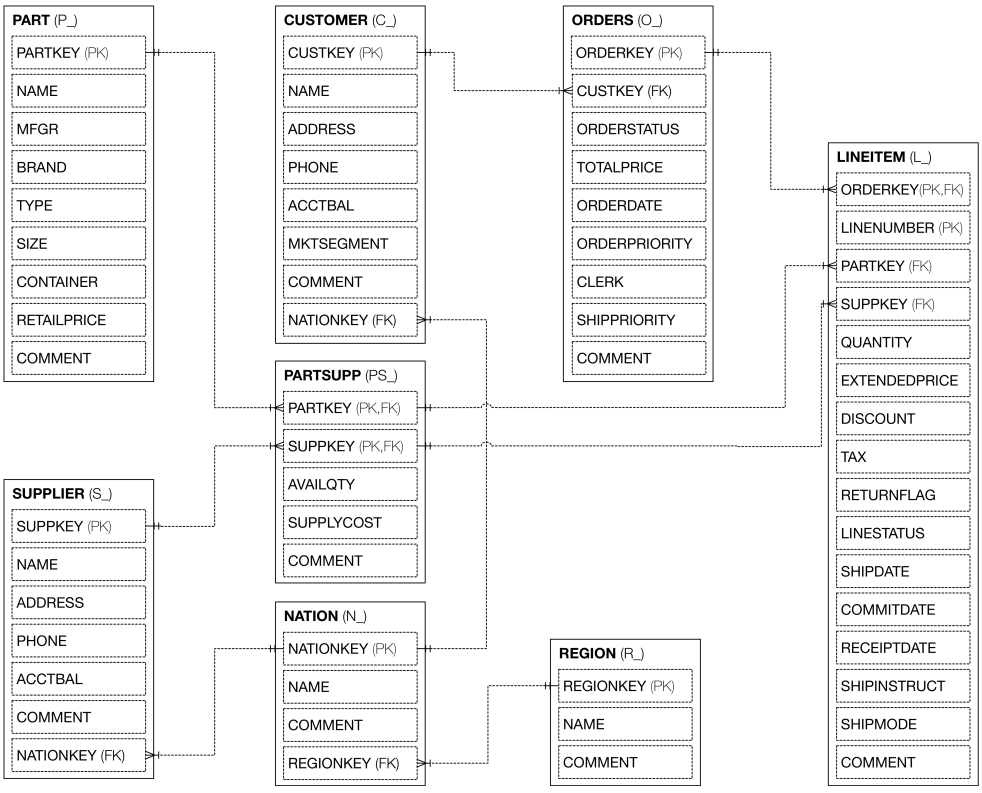

Fig. 9. RDM for the TPC-H data model. Redrawn based on [46].

For the analytical tables (Rule CDM.3.2), $PartSupp_T$ includes five column-families, $PartSupp_{CF}$, $Part_{CF}$, $Supplier_{CF}$, $Nation_{CF}$ and $Region_{CF}$; $Orders_T$ integrates four column-families, $Orders_{CF}$, $Customer_{CF}$, $Nation_{CF}$ and $Region_{CF}$; and, $LineItem_T$ integrates the eight column-families, achieving a complete denormalization of the relational schema: $LineItem_{CF}$, $PartSupp_{CF}$, $Part_{CF}$, $Supplier_{CF}$, $Orders_{CF}$, $Customer_{CF}$, $Nation_{CF}$ and $Region_{CF}$. Moreover, and for this table, it is worth mentioning that $Nation_{CF}$ and $Region_{CF}$ are denormalized twice, as there is the need of specifying the nation and region of the supplier, and the nation and region of the customer.

After following the proposed rules, the identified *CDM* for HBase includes five descriptive tables, as depicted in Fig. 11 showing the proposed keys (Rule CDM.4) and columns. For the analytical tables, and also showing the proposed keys (Rule CDM.4) and columns, Fig. 12 highlights the three obtained analytical tables and the virtual aggregation of column-families to verify the several denormalizations of $Nation_{CF}$ and $Region_{CF}$.

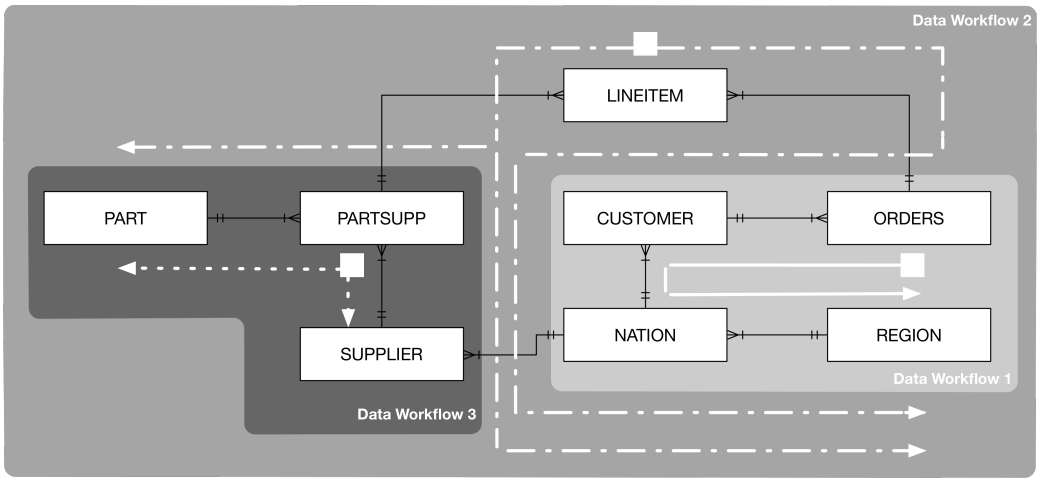

Fig. 10. Data workflows for the TPC-H data model.

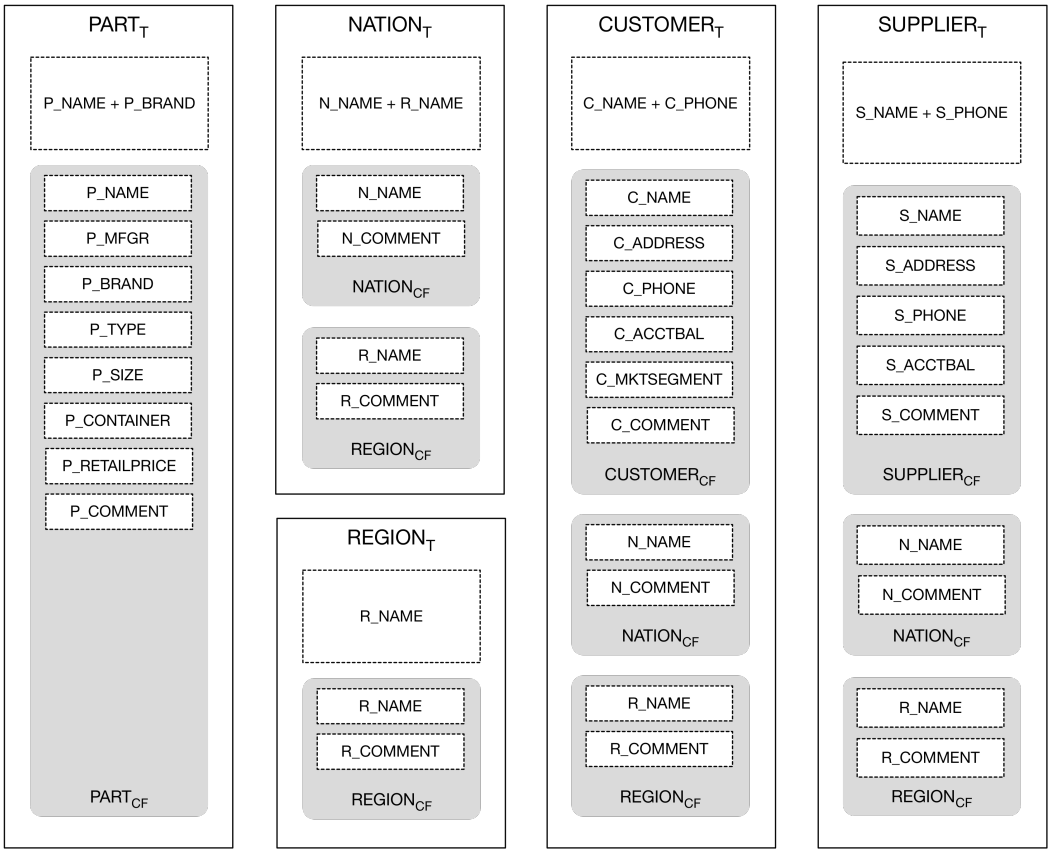

Fig. 11. *CDM* with the descriptive tables for the TPC-H data model.

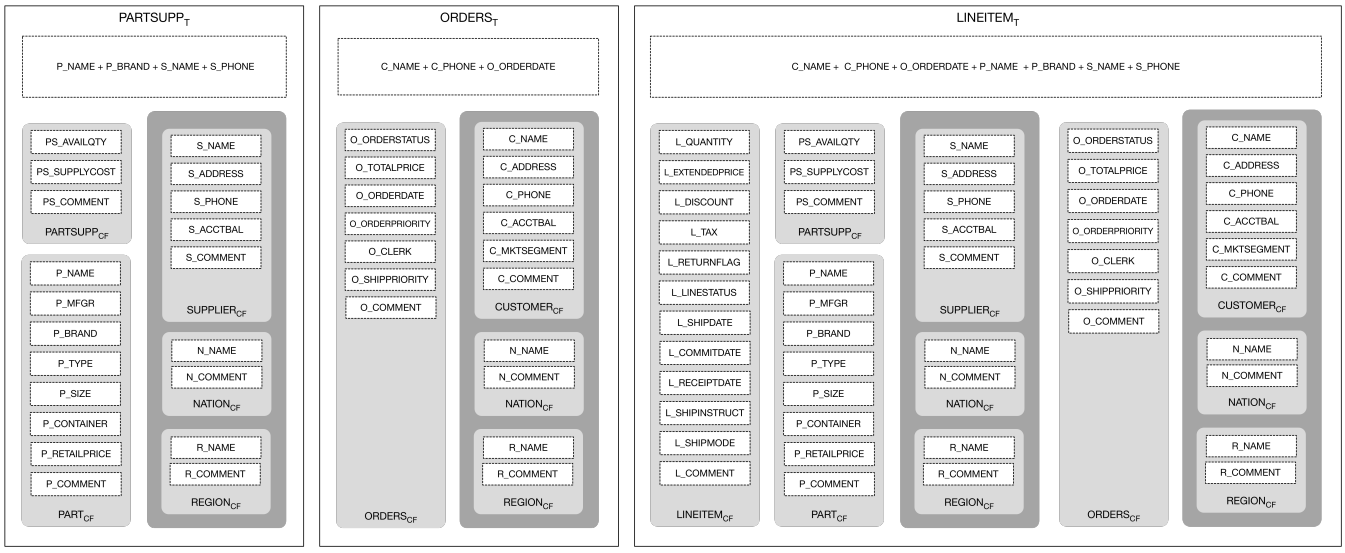

Fig. 12. *CDM* with the analytical tables for the TPC-H data model.

After following the proposed rules for transforming a *RDM* into a *CDM*, several tables are obtained, addressing different purposes in terms of data management. Some tables have a more operational role, maintaining information about the main entities of an application domain, and some others present an analytical profile, being able to answer specific questions about the business processes of an organization. It is up to the information system engineers to decide which tables must be implemented, having into consideration the data management and analytical tasks that must be supported. In particular, for the analytical tables, $PartSupp_T$ and $Orders_T$ are fully contained in $LineItem_T$, so all queries can be answered by this last one, being redundant the implementation of the other two tables.

Another issue that depends of the business context in which these tables are going to be used is the definition of the tables' keys. The keys here presented are merely suggestions of how they can be composed and defined for each table. For instance, these can be more appropriate for inserting or searching data, depending on the attributes and concatenation order between them.

V. Big Data Challenges

This section presents several challenges regarding Big Data, including general dilemmas, challenges in the Big Data life cycle, issues in security, privacy and monitoring, and required changes in organizations. These challenges are also useful for identifying relevant research topics in this field.

A. Big Data General Dilemmas

General dilemmas may include challenges such as the lack of consensus and rigor in Big Data's definition, models and architectures, for example. In [6], the authors claim that the concept of Big Data is often more commercial speculation than it is a scientific research topic. The authors also mention the lack of standardization in Big Data, such as data quality evaluation and benchmarking. In fact, the lack of standard benchmarks to compare different technologies is seriously aggravated by the constant technological evolution in Big Data environments [47].

Even the ways to fully use Big Data remain an open subject to explore, such as applications in science, engineering, medicine, finance, education, government, retail, transportation or telecommunications, for example [6]. Discussions such as how to select the most appropriate data within several sources, or how to estimate their value, remain as Big Data dilemmas [19]. Another commonly discussed pitfall is how Big Data helps representing the population better than a small dataset [22]. This obviously varies with the context, but the authors call our attention for not assuming that more data is always better.

B. Challenges in the Big Data Life Cycle

These challenges are related to technical difficulties in tasks such as Big Data collection, integration, cleansing, transformation, storage, processing, analysis or governance:

- The need to rethink storage devices, architectures, mechanisms and networks, in order to achieve more efficient input/output (I/O), data accessibility and data transmission [10];
- Scalability becomes crucial to store and analyze data. Handling increasing amounts of data requires redesigning databases and algorithms to extract value from it [27]. Parallel computing becomes crucial to deal with Big Data, assuring availability, cost efficiency and elasticity [6];
- Assuring data quality and adding value through data preparation becomes challenging in Big Data environments [10]. Different data sources may have different data quality problems [27]. These problems and vast amounts of redundancy can also make data integration more difficult [6]. The heterogeneity resulting from multiple sources augment these challenges, as traditional techniques for data analysis expect homogeneous data [48]. Heterogeneity brings implications on data integration [28] and consequences in the analysis of Big Data, since the unstructured nature of data sources presents several challenges regarding transformations to support adequate analytical tasks;

- Visualizing Big Data requires rethinking traditional approaches due to the volume of data, thus joining appearance and functionality is crucial [10]. Advanced data visualizations are needed to extract value from Big Data [49], having the capability to scale to thousands or millions of data points, handle multiple data types, and be easy to use in order to satisfy several users. Manipulating Big Data is challenging due to its characteristics, namely executing drilldowns or rollups. In these visualizations, data from multiple sources is typically integrated into a single picture. Technological evolutions are being made to address the challenge of manipulating Big Data interactively [14];
- Searching, mining and analyzing Big Data is a challenging and relevant research trend, including Big Data searching algorithms, recommendation systems, real-time Big Data mining, image mining, text mining, among others [6]. As [15] claim, size is frequently the main concern in Big Data, but the unstructured nature of data also deserves attention (e.g., text, audio, video) and imposes significant challenges in these tasks;
- Big Data governance faces challenges regarding control and authority over massive amounts of data from different sources [27]. Managing such heterogeneous environment to plan access policies and assure traceability can quickly become almost impossible without adequate governance tools.

Organizations face several challenges in Big Data life cycle. New business problems require technological innovations in the way data flows across the organization. Surpassing these challenges will depend on the organization's maturity, since legacy applications and the use of incompatible formats can impose several difficulties to an adequate integration and extraction of value from Big Data. Collecting data, namely gaining access to it, may also be a challenge, since integrating data from multiple sources, including external ones, raises questions about others' intention to share it free of charge [5].

In [6], the authors state that the efficiency in data flows is key to assure an adequate Big Data processing. The authors also highlight the challenge of building effective computing models in real-time and online applications to analyze Big Data. Other challenges related to processing Big Data may include reutilization and reorganization of data, which become laborious at scale. The characteristics of Big Data require a paradigm shift in databases and analytical technologies, since dealing with Big Data throughout its life cycle can potentially create severe bottlenecks in networks, storage devices and relational databases. Technology is evolving to execute these stages in distributed environments, becoming dependent on high storage capacity and processing power.

Even relational databases are evolving to accommodate these trends, increasing query performance and being able to deal with more data variety [3]. Combining the benefits of RDBMSs and NoSQL databases actually represents a research trend, as well as query optimization in Big Data technologies [28]. Furthermore, advancements are being made in scalable storage and algorithms. The work of [50] argues that processing queries in Big Data may take significant time, since it is challenging to sequentially iterate through the whole dataset in a short amount of time. Consequently, the authors highlight the relevance of designing indexes and consider adequate preprocessing technologies. In [27], it is identified the need to study adequate models to store and retrieve data, a crucial factor to successfully implement Big Data solutions. Models and algorithms for scalable data analysis also remain an open research issue, as well as the integration and analysis of data arriving continuously from streams. Mining data streams has been identified as an emergent research topic in Big Data analytics [51].

C. Big Data in Secure, Private and Monitored Environments

Nowadays, keeping data secure and private is one of the most concerning tasks for organizations [6], [48]. Users want to rest assure that any leaks into the public domain will not occur [19]. In [52], citing a survey from [53], it is claimed that security and privacy are frequently mentioned among the Big Data concerns of Information Technology (IT) managers. It is relevant to plan a Big Data driven security model for organizations to accurately specify risks and prevent illegal activity or cyber threats. Several considerations are mentioned, such as authentication, authorization, network traffic analysis, data protection laws and mining data related to security. In [6], it is also discussed the potential for Big Data applications related to security concerns.

Due to the characteristics of Big Data, more risks arise and traditional data protection methods must be rethought. The work of [6] argues that Big Data applications face multiple challenges related to security, privacy and monitoring: protection of personal privacy during not only data collection, but also in its subsequent storage and flows; Big Data quality and its influence to appropriate and secure uses; the performance of security mechanisms like encryption is largely influenced by the scale and variety of data; and other aspects related to secure communications, administration and monitoring in environments with multiple users and services. Other relevant challenge, as highlighted by [27], is assuring Big Data integrity, i.e., data is only modified by the owner or other authorized entities.

Policies related to data are also relevant today, at a time when there is a significant amount of sensitive data about individuals, such as the one related with their health or finances [5]. Legal issues are being raised regarding the easiness to copy, integrate and recurrently use data by different people. Intellectual property, data ownership and responsibility regarding inaccurate data, deserve proper attention from policy makers [5]. Legal and regulatory issues deserve attention in several aspects [50], like analyzing the adequacy of current laws and regulations to properly protect data about individuals [27]. Even the constant tracking on employees within an organization can raise discussions about adequate work policies [54].

Besides these issues, the work of [55] raises questions about the implications of having data widely and transparently available. Assuring privacy is both a technical

and sociological problem, as [48] argue. Inadequate availability of location-based data allows the possibility to infer a person's residence, office location and identity, for example. Moreover, many other data sources can contain personal identifiers, or even if no personal identifiers exist, when data is rich enough, one can draw reasonable inferences from it [56]. Another relevant topic, briefly mentioned above, is data ownership, due to its value for certain organizations that are currently debating about ways of sharing or selling data without losing control of it [48]. Data ownership is often discussed regarding social media websites, since users' data is not owned by organizations, although they store it [19]. As [56] discuss, these organizations tend to assume that they hold the rights of the data, and sometimes the current legislation benefits them, allowing organizations to not permanently delete data, even when users ask for it.

Discussing Big Data security, privacy and monitoring in cloud environments is also relevant. Organizations frequently recognize that using Big Data technologies in cloud environments helps reducing their IT costs [50], although raising concerns about Big Data storage and processing infrastructures. Therefore, as [50] claim, one of the challenges lies in assuring adequate monitoring and security without exposing users' data when processing it.

D. Organizational Change

Surely Big Data may sound appealing to most organizations, but frequently organizational leaders lack the understanding of its value and how to extract it [5]. Occasionally, the lack of knowledge in how to use analytics is mentioned as the leading obstacle to become more data-driven [57]. Within several business areas, organizations need to monitor trends and gain advantages compared to their competitors, but as [5] discuss, many of them lack the talent, the rigorous workflows structure and the incentives for adequate Big Data initiatives to better support decision-making. Leaders and policy makers must understand how Big Data can create value, as well as critically thinking about IT capabilities, data strategies, analytical talent and data-driven approaches.

This paradigm shift in organizations requires them to move analytics into the core business and operational functions [3], changing business processes, delivering insights related to customers, products, services, and other transactions. [58] present challenges that organizations will face in management, caused by Big Data initiatives, among which can be highlighted: the need for adequate leadership and data scientists [59], computer scientists or other professionals to deal with Big Data; the adequate understanding and use of Big Data technology; and the need to change organizational culture. Big Data initiatives require a multidisciplinary approach, demanding collaboration to deliver useful results that must be properly understandable by the organization [48], although to accomplish this, significant organizational changes must occur.

VI. Conclusion

This paper presented a state-of-the-art literature review in Big Data and also a modelling approach to convert relational databases into column-oriented databases. Besides this modelling approach, this work discussed the relevance, definition and challenges in Big Data, as well as how to design Big Data solutions through the use of current techniques and technologies.

Big Data is a concept of major relevance in today's world, and its popularity has increased considerably during the last years. Business areas like healthcare, retail, manufacturing and modern cities will benefit from the collection, storage, processing and analysis of Big Data, leveraging unprecedented data-driven workflows and considerably improving decision-making. This new type of data is being defined not only by its characteristics (e.g., volume, variety, velocity), but also by the limitations it imposes in traditional technologies.

Several requirements and techniques to design Big Data solutions were discussed. Moreover, Hadoop and related projects were presented, as well as scalable databases and other technologies for Big Data Analytics. One can conclude that there is no common approach to work with Big Data and there are innumerous technologies to choose from, each trying to stand out, which creates barriers in the design and implementation of Big Data solutions, since most of the time they are competitors, or their role is misunderstood, overlapping each other. Logical architectures discussed in this work solve part of the problem by orchestrating components according to their role and interactions, but ambiguity regarding the Big Data technology most suitable to a specific role still prevails, mainly due to the diverse set of possible choices.

Organizations implementing Big Data initiatives will face many challenges, among which can be highlighted: the lack of consensus in definitions, models or architectures; challenges regarding the Big Data life cycle; concerns related to security, privacy and monitoring; and organizational change, such as new skills and changes in business processes. The proposed modelling approach to convert relational databases into column-oriented databases aims to solve part of these challenges, helping practitioners to perform migrations from traditional contexts to Big Data contexts.

For future work, one will study the concept of Big Data Warehouse (BDW), namely its main characteristics, changes from traditional environments, storage technologies, advancements in OLAP, query and integration mechanisms for BDWs, and implementations in specific contexts.

References

[1] E. Dumbill, "Making sense of big data," *Big Data*, vol. 1, no. 1, pp. 1–2, 2013.

[2] R. L. Villars, C. W. Olofson, and M. Eastwood, "Big data: What it is and why you should care," IDC, Report, 2011.

[3] T. H. Davenport, P. Barth, and R. Bean, "How big data is different," *MIT Sloan Management Review*, vol. 54, no. 1, pp. 43–46, 2012.

[4] Google Trends, "Interest in Big Data over time," 13-Feb-2016. [Online]. Available: https://www.google.pt/trends/explore#q=big%20data. [Accessed: 13-Feb-2016].

[5] J. Manyika *et al.*, "Big data: The next frontier for innovation, competition, and productivity," McKinsey Global Institute, Report, 2011.

[6] M. Chen, S. Mao, and Y. Liu, "Big Data: A Survey," *Mobile Networks and Applications*, vol. 19, no. 2, pp. 171–209, Apr. 2014.

[7] J. S. Ward and A. Barker, "Undefined By Data: A Survey of Big Data Definitions," *arXiv:1309.5821 [cs.DB]*, Sep. 2013.

[8] S. Madden, "From databases to big data," *IEEE Internet Computing*,

vol. 16, no. 3, pp. 4–6, 2012.
[9] C. Costa and M. Y. Santos, "A Conceptual Model for the Professional Profile of a Data Scientist," presented at the WorldCist - World Conference on Information Systems and Technologies, 2017.
[10] C. L. Philip Chen and C.-Y. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on Big Data," *Information Sciences*, vol. 275, pp. 314–347, Aug. 2014.
[11] S. Sakr, F. Bajaber, A. Barnawi, A. Altalhi, R. Elshawi, and O. Batarfi, "Big Data Processing Systems: State-of-the-Art and Open Challenges," in *2015 International Conference on Cloud Computing (ICCC)*, 2015, pp. 1–8.
[12] A. Siddiqa *et al.*, "A survey of big data management: Taxonomy and state-of-the-art," *Journal of Network and Computer Applications*, vol. 71, pp. 151–166, Aug. 2016.
[13] W. Fan and A. Bifet, "Mining big data: current status, and forecast to the future," *ACM SIGKDD Explorations Newsletter*, vol. 14, no. 2, pp. 1–5, 2013.
[14] K. Krishnan, *Data Warehousing in the Age of Big Data*, 1st ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2013.
[15] A. Gandomi and M. Haider, "Beyond the hype: Big data concepts, methods, and analytics," *International Journal of Information Management*, vol. 35, no. 2, pp. 137–144, Apr. 2015.
[16] D. Laney, "3D Data Management: Controlling Data Volume, Velocity, and Variety," META Group Inc, Report, Jun. 2001.
[17] M. Schroeck, R. Shockley, S. Janet, D. Romero-Morales, and P. Tufano, "Analytics: The real-world use of big data. How innovative enterprises extract value from uncertain data." IBM Institute for Business Value, 2012.
[18] P. Zikopoulos and C. Eaton, *Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data*, 1st ed. McGraw-Hill Osborne Media, 2011.
[19] P. Chandarana and M. Vijayalakshmi, "Big Data analytics frameworks," in *2014 International Conference on Circuits, Systems, Communication and Information Technology Applications (CSCITA)*, 2014, pp. 430–434.
[20] N. Marz and J. Warren, *Big Data: Principles and best practices of scalable realtime data systems*. Manning Publications Co., 2015.
[21] L. Garber, "Using In-Memory Analytics to Quickly Crunch Big Data," *Computer*, vol. 45, no. 10, pp. 16–18, Oct. 2012.
[22] D. Fisher, R. DeLine, M. Czerwinski, and S. Drucker, "Interactions with big data analytics," *Interactions*, vol. 19, no. 3, pp. 50–59, 2012.
[23] R. Kimball and M. Ross, *The data warehouse toolkit: The definitive guide to dimensional modeling*, 3rd ed. John Wiley & Sons, 2013.
[24] K. Kambatla, G. Kollias, V. Kumar, and A. Grama, "Trends in big data analytics," *Journal of Parallel and Distributed Computing*, vol. 74, no. 7, pp. 2561–2573, Jul. 2014.
[25] E. Begoli and J. Horey, "Design Principles for Effective Knowledge Discovery from Big Data," in *2012 Joint Working IEEE/IFIP Conference on Software Architecture (WICSA) and European Conference on Software Architecture (ECSA)*, 2012, pp. 215–218.
[26] J. M. Tien, "Big Data: Unleashing information," *J. Syst. Sci. Syst. Eng.*, vol. 22, no. 2, pp. 127–151, Jun. 2013.
[27] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, and S. U. Khan, "The rise of 'big data' on cloud computing: Review and open research issues," *Information Systems*, vol. 47, pp. 98–115, Jan. 2015.
[28] A. Cuzzocrea, I.-Y. Song, and K. C. Davis, "Analytics over large-scale multidimensional data: the big data revolution!," in *Proceedings of the ACM 14th international workshop on Data Warehousing and OLAP*, 2011, pp. 101–104.
[29] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google file system," in *ACM SIGOPS operating systems review*, 2003, vol. 37, pp. 29–43.
[30] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008.
[31] F. Chang *et al.*, "Bigtable: A Distributed Storage System for Structured Data," *ACM Trans. Comput. Syst.*, vol. 26, no. 2, p. 4:1–4:26, Jun. 2008.
[32] NBD-PWG, "NIST Big Data Interoperability Framework: Volume 6, Reference Architecture," National Institute of Standards and Technology, Technical Report NIST SP 1500-6, Oct. 2015.
[33] K. Bakshi, "Considerations for big data: Architecture and approach," in *2012 IEEE Aerospace Conference*, 2012, pp. 1–7.
[34] Apache Hadoop, "Welcome to Apache Hadoop," *Welcome to Apache Hadoop*, 2016. [Online]. Available: https://hadoop.apache.org/. [Accessed: 01-Feb-2017].
[35] Hortonworks, "Solving Apache Hadoop Security: A Holistic Approach to a Secure Data Lake," Hortonworks, White paper, 2016.
[36] K. Grolinger, W. A. Higashino, A. Tiwari, and M. A. Capretz, "Data management in cloud environments: NoSQL and NewSQL data stores," *Journal of Cloud Computing: advances, systems and applications*, vol. 2, no. 1, p. 22, 2013.
[37] R. Cattell, "Scalable SQL and NoSQL data stores," *ACM SIGMOD Record*, vol. 39, no. 4, pp. 12–27, 2011.
[38] E. Brewer, "CAP twelve years later: How the 'rules' have changed," *Computer*, vol. 45, no. 2, pp. 23–29, Feb. 2012.
[39] B. G. Tudorica and C. Bucur, "A comparison between several NoSQL databases with comments and notes," in *Roedunet International Conference (RoEduNet), 2011 10th*, 2011, pp. 1–5.
[40] P. Myers, "Big Data Analytics with PowerPivot and Power View," presented at the Pass SQL Saturday, 2013.
[41] M. Theuwissen, "R vs Python for Data Science: The Winner is …," *KDnuggets*, 2015. [Online]. Available: http://www.kdnuggets.com/2015/05/r-vs-python-data-science.html. [Accessed: 28-Mar-2016].
[42] M. Y. Santos and C. Costa, "Data Models in NoSQL Databases for Big Data Contexts," in *2016 International Conference of Data Mining and Big Data (DMBD)*, 2016, pp. 1–11.
[43] M. Y. Santos and C. Costa, "Data Warehousing in Big Data: From Multidimensional to Tabular Data Models.," in *Ninth International C* Conference on Computer Science & Software Engineering (C3S2E)*, 2016, pp. 51–60.
[44] A. Khurana, "Introduction to HBase schema design," vol. 35, no. 5, pp. 29–36, 2012.
[45] E. Hewitt, *Cassandra: the definitive guide: [distributed data at web scale]*, 1. ed. Beijing: O'Reilly, 2011.
[46] TPC, "TPC-H Specification," 2017. [Online]. Available: http://www.tpc.org/tpc_documents_current_versions/pdf/tpc-h_v2.17.2.pdf. [Accessed: 01-Jun-2017].
[47] C. Baru, M. Bhandarkar, R. Nambiar, M. Poess, and T. Rabl, "Benchmarking Big Data Systems and the BigData Top100 List," *Big Data*, vol. 1, no. 1, pp. 60–64, Feb. 2013.
[48] H. V. Jagadish *et al.*, "Big data and its technical challenges," *Communications of the ACM*, vol. 57, no. 7, pp. 86–94, 2014.
[49] P. Russom, "Big data analytics," TDWI Research, Best Practices Report, 2011.
[50] C. Ji, Y. Li, W. Qiu, U. Awada, and K. Li, "Big Data Processing in Cloud Computing Environments," *Proceedings of the 2012 12th International Symposium on Pervasive Systems, Algorithms, and Networks (i-Span 2012)*, pp. 17–23, 2012.
[51] H. Chen, R. H. Chiang, and V. C. Storey, "Business Intelligence and Analytics: From Big Data to Big Impact.," *MIS quarterly*, vol. 36, no. 4, pp. 1165–1188, 2012.
[52] S. Sagiroglu and D. Sinanc, "Big data: A review," in *2013 International Conference on Collaboration Technologies and Systems (CTS)*, 2013, pp. 42–47.
[53] Intel IT Center, "Peer Research: Big Data Analytics," Intel, Report, Aug. 2012.
[54] K. Michael and K. W. Miller, "Big Data: New Opportunities and New Challenges [Guest editors' introduction]," *Computer*, vol. 46, no. 6, pp. 22–24, Jun. 2013.
[55] B. Brown, M. Chui, and J. Manyika, "Are you ready for the era of 'big data,'" Report, 2011.
[56] M. R. Wigan and R. Clarke, "Big Data's Big Unintended Consequences," *Computer*, vol. 46, no. 6, pp. 46–53, Jun. 2013.
[57] S. LaValle, E. Lesser, R. Shockley, M. S. Hopkins, and N. Kruschwitz, "Big data, analytics and the path from insights to value," *MIT sloan management review*, vol. 52, no. 2, pp. 21–32, 2011.
[58] A. McAfee, E. Brynjolfsson, T. H. Davenport, D. J. Patil, and D. Barton, "Big data. The Management Revolution," *The management revolution. Harvard Bus Rev*, vol. 90, no. 10, pp. 61–67, 2012.
[59] F. Provost and T. Fawcett, "Data Science and its Relationship to Big Data and Data-Driven Decision Making," *Big Data*, vol. 1, no. 1, pp. 51–59, 2013.

Carlos Costa, MSc, is an invited lecturer and PhD student in Information Systems at the University of Minho. Previously, his academic formation was the Master's in Engineering and Management of Information Systems at the University of Minho. His previous professional experience includes: web designer at QUATIC 2014; web developer and administrator at APSI/PTAIS; president at AIS.SC UMinho; software developer as a freelancer; database administrator and project manager at Conde Parish; and computer technician at Oversystems. His current R&D interests include: Data Science (data mining, data cleansing, data enrichment, and

data visualization); Big Data; Data Warehousing; OLAP; OLTP databases (SQL and NoSQL); software development; and Information Systems architectures. Mr. Carlos won the best paper award in the IAENG ICDMKE'15 conference.

Maribel Yasmina Santos, PhD, Habilitation, is an Associate Professor at the Department of Information Systems, University of Minho, Portugal. Maribel received the Aggregated title (Habilitation) in Information Systems and Technologies from the University of Minho in 2012 and a Ph.D. in Information Systems and Technologies from the University of Minho in 2001. She was the Secretary-General of the Association of Geographic Information Laboratories for Europe (AGILE) from May 2013 to June 2015 and Associate Director of the Department of Information Systems, University of Minho, from May 2010 to July 2014. Currently, she coordinates the Business and Location-enhanced Database Systems Research Track of SEMAG (Software Engineering and Management Group) at the ALGORITMI Research Centre, where she is a senior researcher. Her research interests include Business Intelligence and Analytics, Big Data Analytics, (Spatial) Data Warehousing and (Spatial) On-line Analytical Processing. Dr. Santos won the best paper award in the IAENG ICDMKE'15 conference.