# The SusCity Big Data Warehousing Approach for Smart Cities

Carlos Costa
ALGORITMI Research Centre,
University of Minho
Guimarães, Portugal
carlos.costa@dsi.uminho.pt

Maribel Yasmina Santos
ALGORITMI Research Centre,
University of Minho
Guimarães, Portugal
maribel@dsi.uminho.pt

## ABSTRACT

Nowadays, the concept of Smart City provides a rich analytical context, highlighting the need to store and process vast amounts of heterogeneous data flowing at different velocities. This data is defined as Big Data, which imposes significant difficulties in traditional data techniques and technologies. Data Warehouses (DWs) have long been recognized as a fundamental enterprise asset, providing fact-based decision support for several organizations. The concept of DW is evolving. Traditionally, Relational Database Management Systems (RDBMSs) are used to store historical data, providing different analytical perspectives regarding several business processes. With the current advancements in Big Data techniques and technologies, the concept of Big Data Warehouse (BDW) emerges to surpass several limitations of traditional DWs. This paper presents a novel approach for designing and implementing BDWs, which has been supporting the SusCity data visualization platform. The BDW is a crucial component of the SusCity research project in the context of Smart Cities, supporting analytical tasks based on data collected in the city of Lisbon.

## CCS CONCEPTS

• **Information systems~Data Warehouses** • **Information systems~Online analytical processing** • *Information systems~Database design and models* • *Information systems~Data analytics*

## KEYWORDS

Big Data, Big Data Warehousing, Data Warehouse, Hadoop, NoSQL, Smart Cities



IDEAS '17, July 12-14, 2017, Bristol, United Kingdom
© 2017 Association for Computing Machinery.
ACM ISBN 978-1-4503-5220-8/17/07…$15.00
http://dx.doi.org/10.1145/3105831.3105841

**ACM Reference format:**

C. Costa, M.Y. Santos. 2017. The SusCity Big Data Warehousing Approach for Smart Cities. In *Proceedings of International Database Engineering & Applications Symposium*, Bristol, United Kingdom, *July 2017 (IDEAS'17)*, 10 pages.
DOI: 10.1145/3105831.3105841

## 1 INTRODUCTION

Advancements in Information and Communication Technologies (ICT) allow for things and humans in our world to be constantly interconnected [62]. Nowadays, people use several devices as an extension of their cognitive capabilities, generating huge amounts of heterogeneous data at ever-increasing rates. In the context of Smart Cities, data is constantly being produced by an extensive network of interconnected things, including smartphones, smart meters, temperature sensors, noise sensors, smart appliances, location sensors, among many others. Moreover, there are also other data sources such as the city's transactional database systems, geospatial files, census data, and data provided by private companies responsible for certain city services. This phenomenon is typically associated with the concepts of Internet of Things (IoT) and Big Data [26]. Consequently, Smart Cities are seen as rich Big Data analytics contexts, given these extensive set of data sources and their relevance in the city's decision-making process.

Big Data can be defined as data whose volume, variety and velocity impose significant challenges in traditional techniques and technologies [36, 60], such as a traditional relational-based DW, which has several limitations, including the fact that it is less scalable than emergent Big Data technologies and costs are typically higher [33]. Relational DWs are also modeled according to rigid data modeling techniques, often ineffective/inefficient for streaming contexts [17] and for storing huge amounts of unstructured or semi-structured data, including text, video, images, GeoJSON data, and nested structures like maps or arrays. In the last years, research and development in Big Data made openly available new techniques and technologies for practitioners, namely: distributed file systems, such as the Hadoop Distributed File System (HDFS) [49]; distributed data processing paradigms, like Hadoop MapReduce [13] and Spark [48]; and NoSQL databases, such as Cassandra, HBase or MongoDB [3].

Taking into consideration the limitations of traditional DWs and the scarce scientific contributes regarding Big Data Warehousing, this work presents a novel approach for designing and implementing BDWs, assuring the following characteristics:

parallel/distributed processing [28, 31, 33]; mixed and complex analytical workloads, including experimental/exploratory analysis (sandboxing) [19, 33, 38], advanced simulation [19], reporting, ad hoc querying [30], computation of materialized views [17], data mining, text mining and visualization [33]; flexible storage to accommodate structured, semi-structured and unstructured data [31, 33, 38]; stream processing, low latency and high frequency updates [15, 17, 38, 45]; high performance analytics with near real-time response [38, 44, 45]; scalability, supporting ever-increasing data volumes, users and analytical workloads [44]; and commodity hardware to reduce the costs of maintenance and scalability [33].

A BDW built using the proposed approach is of major relevance for organizations, not only supporting fact-based decision-making, but also providing streaming and predictive capabilities, even if data is semi-structured or unstructured. This work provides a Big Data Warehousing approach that practitioners can use as a guide for implementing these complex systems. It includes both the logical layer (data model, logical components and data flows) and physical layer (technological infrastructure) [45]. This approach is a scientific output focusing on the foundations for Big Data Warehousing, and is being developed using the Design Science Research Methodology for Information Systems [22, 40]. The SusCity project [52] in the context of Smart Cities represents one of the proof-of-concept implementations aiming to evaluate the effectiveness and efficiency of this approach. Despite the fact that this approach is here discussed in the context of Smart Cities, it can be generalized and, therefore, be used to implement BDWs in other contexts.

This paper is organized as follows: section 2 presents the work related to Big Data Warehousing in Smart Cities; section 3 presents the SusCity Big Data Warehousing approach for designing and implementing BDWs; section 4 presents the data modelling strategy associated with the SusCity BDW; section 5 presents the SusCity data visualization platform; and section 6 concludes with remarks about the undertaken work and some considerations for future work.

## 2 RELATED WORK

Nowadays, the community is studying the role of the DW in Big Data environments [33] and, therefore, the concept of BDW is emerging [38]. Some works explore implementations of DWs on top of NoSQL databases, such as document-oriented [6, 7], column-oriented [7] and graph databases [20], despite the fact that they were mainly designed for scaling Online Transactional Processing (OLTP) applications [3]. There are other works focusing on the storage technologies and optimizations, discussing SQL-on-Hadoop systems like Hive [55] and Impala [32], or improving these technologies through new storage and processing mechanisms [25]. Furthermore, some authors propose advancements in analytical and integration mechanisms suitable for BDWs [34, 51, 57]. Finally, other works discuss implementations of certain technologies in specific contexts, which can be related to certain characteristics of a BDW, such as the Data Warehousing infrastructure at Facebook [54] or the use of NoSQL DWs in medicine [59].

Currently, the state-of-the-art shows that the design of BDWs should focus both on the physical layer (technological infrastructure) and logical layer (data model, logical components and data flows) [45]. In general terms, the BDW can be implemented using two strategies. The first is the "lift and shift" strategy [8], augmenting the capabilities of relational DWs with Big Data technologies, such as Hadoop or NoSQL, in order to solve specific use cases. This represents a use case driven approach instead of a data modelling approach, which often leads to possible uncoordinated data silos [8]. The second is the "rip and replace" strategy, in which a traditional DW is completely replaced by Big Data technologies [44, 45].

Other scientific contributions in the topic of Big Data in Smart Cities can also be related to the work proposed in this paper. [56] discusses a conceptual model to support several services in Smart Cities contexts, including concepts related to Big Data processing, Open Data, web services, data streams, among others. [29] proposes an approach for Big Data analytics-as-a-service in Smart Cities. [16] proposes an architecture to collect and process data from sensors. [50] presents a cloud-based Big Data platform for Smart Cities. [5] shared the Big Data architecture designed for the SmartSantander initiative, based on CouchDB [1] to support Smart City applications. In the architecture proposed in [5], CouchDB, which is a document-oriented NoSQL database, is fueled using data gathered from sensors and other external sources.

In previous works, we already addressed the topic of general Big Data architectures to store and process data in Smart Cities contexts, proposing new services based on clustering and time series forecasting using Big Data technologies [11]; studying the role of NoSQL databases in Smart Cities [12]; and proposing a general purpose Big Data architecture for Smart Cities, named BASIS, with different abstraction layers, from the most conceptual to the most technological [10].

The Big Data architectures discussed above serve broader contexts, not including the level of detail specifically required to build BDWs. Consequently, this work aims to support a more focused context, namely the Big Data Warehousing context, in which data is collected, prepared, enriched, stored and processed specifically for querying and Online Analytical Processing (OLAP) over huge amounts of data, preferably through distributed SQL interfaces, in order to support analytical dashboards including historical, real-time, simulated and predictive data, through a platform targeted for decision-makers. The SusCity Big Data Warehousing approach represents a novel approach to design and implement BDWs based on the "rip and replace" strategy [44, 45], which is here explored in the context of the SusCity research project [52]. The lack of integrated and validated Big Data Warehousing approaches is a significant gap in the scientific community, and to contribute to fill some of these gaps, in this paper, we focus on several aspects of a BDW: the logical components of the system; the technologies used to instantiate these components; the infrastructure in which the BDW system can be deployed; and the data modelling strategy,

which is seen as a significant scientific challenge and opportunity in Big Data Warehousing. Consequently, the SusCity Big Data Warehousing approach can help practitioners implementing BDWs in the context of Smart Cities, adequately supporting the needs of the city's decision makers through the value extracted from huge amounts of data, with different types, gathered from several sources and at different rates. The approach proposed in this paper was validated in the context of the SusCity project [52], but can also be used in other implementation contexts besides Smart Cities, and it can help fostering future research in this topic.

## 3 THE SUSCITY APPROACH FOR BIG DATA WAREHOUSING

In the context of Smart Cities, an adequate Big Data Warehousing approach is crucial to support the decision-making process at scale, respecting the characteristics of a BDW enumerated in section 1. Fig. 1 presents the SusCity Big Data Warehousing approach, focusing on the logical and physical layers. The logical layer helps researcher and practitioners understanding the logical components of the system and how data flows throughout these components. It uses part of the

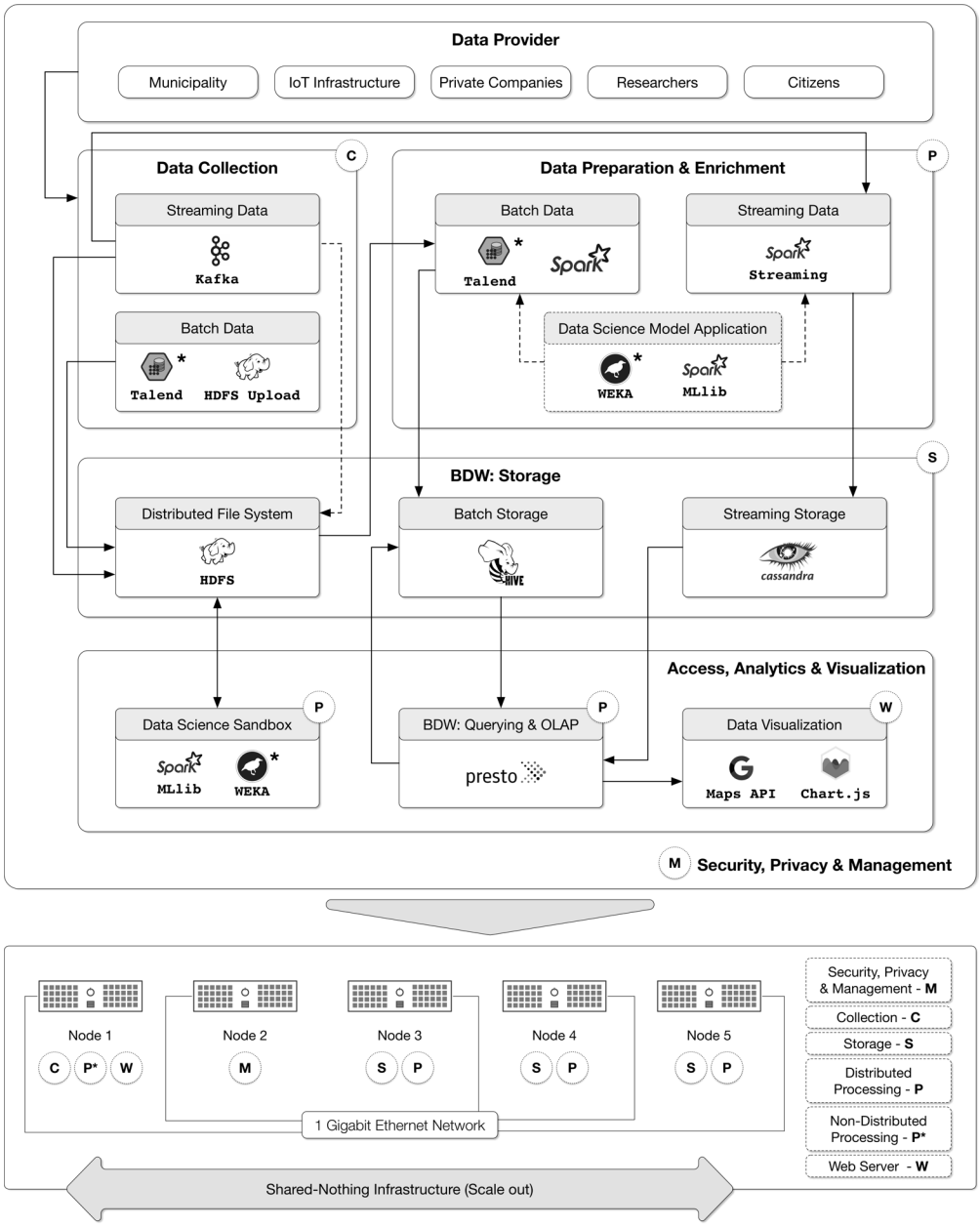


**Figure 1: SusCity Big Data Warehousing approach.**

taxonomy of the Big Data Reference Architecture from the National Institute of Standards and Technology (NIST) [39], as the lack of concepts standardization can be an issue in Big Data research. The physical layer focuses on the technology used for instantiating the logical components and on the infrastructure in which these technologies are deployed.

## 3.1 Logical Components and Data Flows

Regarding the logical layer, the first component is the data provider, which makes data available for further storage and processing. In a typical Smart Cities context, which is also the case of the SusCity research project [52], the data provider can be represented by several actors:

- Municipality – the municipality itself is able to make available several data sources relevant for analytical tasks, including buildings information or geospatial representations of the city's infrastructures, for example. The city's transactional systems can also be valuable data sources;
- IoT infrastructure – includes different kinds of sensors reporting electricity consumption, temperature, noise and mobility patterns, for example. This data is significantly relevant to understand events and real-time patterns in the city;
- Private companies – the city's infrastructures are not always public and, therefore, interactions with private companies are of major relevance in Smart Cities, in order to collect historical energy consumption data, buildings certificates, water consumption, census data, among many other data sources;
- Researchers and citizens – research projects being conducted in the city are a relevant data source for the BDW, including simulation data regarding different phenomena in the city (e.g., buildings' energy efficiency and mobility patterns), or any other relevant insights corresponding to scientific studies impacting the city. Moreover, citizens engaged in the initiatives promoted by researchers or the municipality can provide useful data for the decision-making process, such as personal energy consumptions, mobility patterns and service consumption habits.

Taking into consideration the data sources presented in Fig. 1 and discussed above, data may arrive at the BDW via batch or streaming mechanisms. For data arriving in batches, we use Talend Open Studio for Big Data [53] and a HDFS client to upload it to the distributed file system. Before any preparation process, data is first uploaded to HDFS, since raw data may serve further analytical purposes (e.g., training and testing data mining models) or may be useful for disaster recovery, in case some problems occur in the batch storage component. HDFS is capable of handling large amounts of structured, semi-structured and unstructured data, distributing them across several nodes in the cluster for further preparation and enrichment. For data arriving in a streaming fashion, Kafka [2] is used to assure highly scalable and robust data collection. Periodically, we can optionally move data from Kafka to HDFS, using systems such as LinkedIn's Gobblin [43], in cases where streaming raw data is also useful for training and testing data mining models.

To prepare and enrich batch data, the SusCity Big Data Warehousing approach takes into consideration the volume of data. If the dataset being processed fits in the constraints of non-distributed technologies, Talend Open Studio for Big Data is used to prepare and enrich data, since it offers a wide set of processing components (e.g., filtering, aggregation, joins, type parsing) in a user-friendly graphical interface. However, when the volume of the dataset requires distributed processing, Spark [48] is used to accomplish the preparation and enrichment task. In streaming scenarios, we can use Spark Streaming to process data as it arrives at the BDW system. Previously trained data mining models can also be applied in this phase, using WEKA [21] for small-scale algorithms (e.g., classification/regression of previously aggregated data or time series forecasting problems) and Spark MLlib for large-scale algorithms, i.e., when the training set contains huge amounts of very detailed data, not previously aggregated. Relevant data mining use cases in Smart Cities contexts may include forecasting and segmenting energy consumption [11]; predicting attendance in the city's events; or segmenting buildings according to their characteristics and efficiency. The same logic applies to unstructured data, since text mining algorithms, for example, can also be applied using WEKA, Spark or any other suitable technology, in order to extract structured patterns to be further stored in the BDW.

Once data is prepared and enriched, it is stored in the storage component, the BDW itself. The storage component consists of three subcomponents. The distributed file system (HDFS) acts as a staging area and sandbox, storing raw batch data, raw streaming data and temporary files needed in the data science sandbox component. It is a crucial component to assure a flexible storage capable of handling data variety from several sources. HDFS is also the underlying storage system of Hive, the technology used in the batch storage component of the proposed architecture. Hive is a Data Warehousing system on Hadoop, frequently mentioned as the de-facto SQL-on-Hadoop solution. In the SusCity Big Data Warehousing approach, Hive tables (stored as ORC files [25]) are used to store large amounts of structured data, using a novel and flexible data modelling approach we have been working on [46, 47], and discussed in section 4. Hive only stores data arriving in batches, since it is designed to be a fast sequential access system. For fast random access, Cassandra is used as the NoSQL database supporting the streaming storage component, since we can assure hundredths or thousands of concurrent writes required by typical streaming scenarios. Previous benchmarks reveal that Cassandra is a suitable distributed database for intensive random read and random write scenarios [12]. Moreover, Cassandra column families are also modelled according to the approach discussed in section 4.

Despite the current advancements regarding Hive's streaming API and ACID transactions support [23], Hive is not yet suitable for a large number of concurrent writes, generating several small files in HDFS, which can become a bottleneck in

the NameNode. As mentioned, Hive is a Data Warehousing system on Hadoop, which is mainly used to scale OLAP applications. Since NoSQL databases are mainly designed to scale OLTP applications [3], they are not as effective and efficient as Hive in sequential access scenarios, typically required in OLAP applications, as can be further seen in subsection 4.2. Therefore, the SusCity BDW takes into account these trade-offs and maintains two separate storage technologies for batch and streaming data. As techniques and technologies evolve, the same technology may be able to adequately support both scenarios. For example, Kudu is a promising technology aiming to provide a middle-ground between fast sequential access and fast random access [35].

The goal of the BDW is to support analytical tasks. Consequently, the access, analytics and visualization component is crucial to deliver adequate insights for data-driven decision-making processes. The querying and OLAP component, using Presto, assures the communication between the batch storage, the streaming storage, and the data visualization component. Presto was open-sourced by Facebook, and it is seen as a SQL-on-Hadoop system providing low-latency query execution over large amounts of data [42]. In fact, it is more than a SQL-on-Hadoop system, since it can provide a SQL interface to a vast set of storage technologies besides Hive (Hadoop), including NoSQL databases like Cassandra and MongoDB. Therefore, in the proposed architecture, Presto is used to query Hive tables and Cassandra Column families. As previously discussed, since Cassandra is less efficient than Hive for fast sequential access (results in subsection 4.2), we also use Presto to transfer data between Cassandra and Hive, avoiding the accumulation of huge amounts of historical data in the Cassandra column families. For more complex queries that surpass the interactivity threshold defined for the SusCity data visualization component (10 seconds), Presto is also used to create materialized views stored in Hive tables.

Although several improvements have been made in Hive, such as the Tez execution engine [14, 25], Presto achieves significantly faster execution times when querying Hive tables, reason why it is used as the querying and OLAP engine in the proposed architecture. Interactive query execution is one of the main requirements of the SusCity data visualization component, in order to engage users through a responsive interface. Therefore, the data visualization component (discussed in section 5) uses Presto to submit SQL queries to the batch and streaming storage systems. Presto is also able to combine data from these two components using a single query (e.g., joins and unions), which is of major relevance to combine historical and streaming data into a unified "picture". Although some benchmarks demonstrated that interactive SQL-on-Hadoop systems similar to Presto (e.g., Impala) may struggle with datasets that do not fit in memory [14], we do not yet feel the need to use the Hive execution engine, since Presto was able to execute all workloads requested by the testbed of the SusCity project. However, if certain scalability issues arise, the Hive execution engine is always available for more demanding workloads. Scalability will certainly not be an issue, since Presto is being used at Facebook to perform queries over their petabyte scale Hive DW, thus it is possible to scale the cluster to accommodate growing data in a Smart Cities context.

Still concerning analytical tasks, the application of data science models (e.g., data mining and text mining models), is only possible with an adequate sandbox where data scientists can "play" with the data, training and testing models to support their hypothesis [9]. Therefore, the proposed architecture includes a dedicated area, named data science sandbox, which interacts with HDFS. Since raw batch and streaming data can be stored in HDFS, data scientists can interact with this data to produce models capable of extracting patterns and making predictions when new data arrives at the preparation and enrichment component. WEKA and Spark are the driving forces for this purpose, as previously discussed.

Security, privacy and management is a relevant component in the SusCity Big Data Warehousing architecture. There are certain Hadoop-related technologies that can be used to assure a secure environment that is properly managed. Kerberos can be used for secure authentication in Hadoop. To assure an extra-layer of security and privacy, Ranger can be used to deploy rigorous authorization policies, defining which users have access to certain files or tables [24]. Regarding Cassandra's security, TLS/SSL encryption can be used for client-to-node or node-to-node communications. Cassandra also makes available simple password authentication and an internal authorization model. In a Smart Cities context, data privacy is a main concern and, therefore, whenever possible, we encourage the anonymization of sensitive data before storing it in the BDW [10]. Finally, Ambari can be used to manage/monitor the Hadoop components [61].

## 3.2 Infrastructure

All the components and technologies discussed in subsection 3.1 are deployed in 5 commodity hardware machines installed on premises, which have been adequate to support the workloads demanded by the SusCity testbed, including several Gigabytes of data from energy grid simulations, buildings information, geospatial files, historical energy consumption data and more than one hundred smart meters. Each machine has 16GB of RAM, Intel i5 quad-core CPUs and 500GB 7200rpm hard disks (except node 1, which has an Intel i7 quad-core CPU and a 256GB SSD drive).

All the machines are connected using a 1 Gigabit Ethernet switch and 5 Ethernet CAT6 cables. Hadoop clusters should be deployed using at least a 1 Gigabit Ethernet network [49]. Using Big Data technologies, like Hadoop, Spark and Cassandra, which rely on commodity hardware and shared-nothing infrastructures, allows scaling the cluster as data volume increases and workloads become more demanding.

Due to resource limitations, Hadoop and Cassandra nodes are co-located (node 3, node 4 and node 5). Presto and Spark are also deployed in these nodes. Distributed processing technologies should be co-located with storage nodes, since in Big Data environments the processing should be brought closer to the storage, in order to avoid moving large amounts of data through

the network [41]. The collection technologies (Kafka, Talend Open Studio and HDFS client), non-distributed processing technologies (Talend Open Studio and WEKA) and the Web Server (serving dashboards with Chart.js [4] and Google Maps API [18]) are all deployed within node 1, again due to resource limitations. Ideally, and in a production environment, these should have dedicated nodes and Kafka should be distributed across several nodes in the cluster. Node 2, besides being the Hadoop NameNode, assures several tasks related to the security, privacy and management of the cluster, containing the Kerberos Key Distribution Center, Ambari and Ranger, for example.

# 4 THE DATA MODELLING STRATEGY FOR THE SMART CITY'S BIG DATA WAREHOUSE

As previously highlighted, Big Data Warehousing approaches should focus both on the logical layer and physical layer. In section 3, the logical components, data flows and technological infrastructure were presented. In this section, we focus on another logical aspect of the Big Data Warehousing approach proposed in this paper, the data modelling strategy. To adequately discuss the proposed data modelling strategy, Fig. 2 should be taken into consideration, which represents an extract of the data model of the BDW implemented in the SusCity research project. Due to security and privacy issues, the entire list of tables and attributes cannot be presented. Nevertheless, this extract is sufficient to explain to researchers and practitioners the data modelling strategy here presented, extending our previous works on this topic [46, 47]. As can be seen in Fig. 2 and as discussed in the section 3, there are two storage components: one for data arriving in batches and another for data arriving in a streaming fashion. However, the proposed data modelling strategy considers the same concepts for both types of storage.

## 4.1 Analytical Objects and Related Concepts

The main concept in this strategy is the analytical object, representing a subject of interest to be analyzed. Typical analytical objects in Smart Cities may include: general indicators about buildings; buildings energy consumption; losses in the energy grid; indicators about the nodes in the energy grid; and the energy consumption recorded by smart meters.

Making the analogy to traditional DWs, analytical objects have the same capabilities of fact tables. In contrast to fact tables, they are fully denormalized structures, in which all the attributes needed to analyze the subject of interest are included in one single analytical object, without the need for dimension tables, avoiding constant and complex join operations. Join operations in Big Data environments are costly [14, 37, 39, 58], since tables may store huge amounts of data. Joining several dimensions with fact tables for each query can be significantly resource-demanding.

Each analytical object contains two types of attributes: descriptive attributes (top half of analytical objects in Fig. 2) and analytical attributes (bottom half of analytical objects in Fig. 2). Descriptive attributes support typical OLAP tasks such as aggregations and filtering. These are analogous to the attributes of the dimensions in traditional DWs and allow the interpretation of the analytical attributes through different perspectives. Analytical attributes are an analogy to the facts in a traditional fact table, but with the particularity that they can

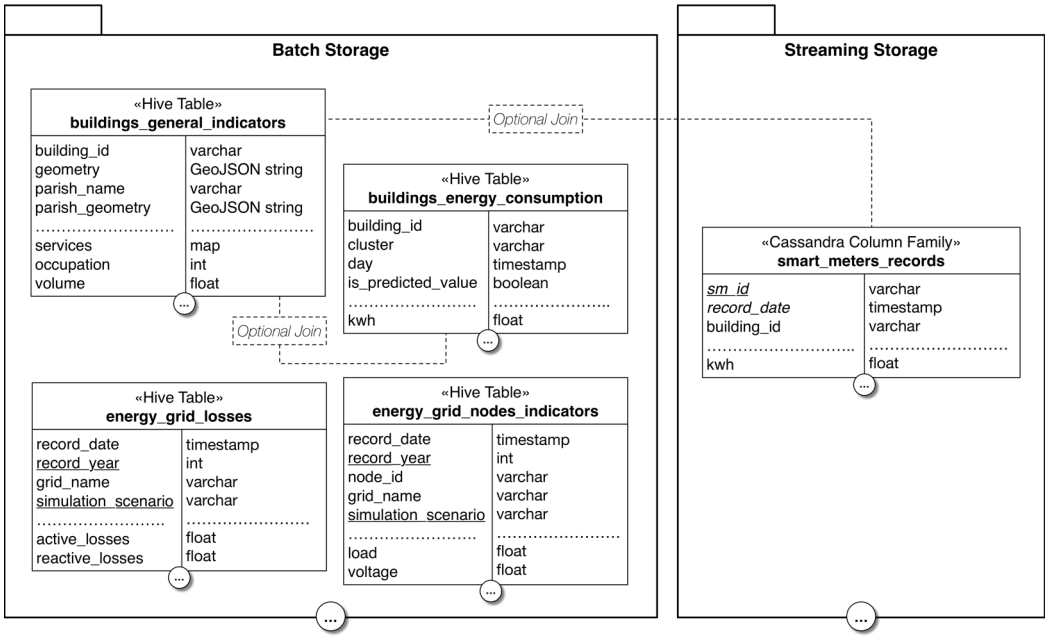


**Figure 2: Data model of the SusCity BDW.**

not only contain facts (historical indicators) but also predictions derived from the application of data mining algorithms. Take as an example the analytical object *"buildings energy consumption"* in Fig. 2, which contains a cluster defining the consumption behavior of each building and a forecast (kWh) of its energy consumption for the following days, information obtained using the WEKA's clustering and time series forecasting algorithms, as proposed in [11].

Descriptive and analytical attributes can store simple data types (e.g., integer, float, varchar) or complex types (e.g., arrays, maps and GeoJSON strings). In the SusCity BDW, complex types are used to store nested structures that will be after interpreted by the data visualization component. For example, a large building can be associated with more than one service (e.g., laundry, supermarket, restaurant, gym). Using a nested complex type like a map, we can store this data using a single record associated with that building. Saving geometry objects in GeoJSON strings is also relevant for geospatial analysis in the Smart Cities context.

Descriptive attributes are also relevant to define partition keys. Certain descriptive attributes can be used to control certain aspects of data locality. In Hive, a partition key distributes the data throughout different folders according to the value of the attribute. In Cassandra, the partition key helps determining which node should be used to store/read the data, since it tries to evenly spread the data across different nodes in the cluster. There is no rigorous rule for defining partition keys and we should evaluate the patterns of the queries and/or the refreshing rates of the analytical objects. For example, in the SusCity BDW, we use the *"simulation scenario"* and the *"record year"* attributes for the partition key in the *"energy grid losses"* and *"energy grid nodes indicators"* Hive tables, since we load data in batches corresponding to yearly simulations for each stress scenario in the energy grid. Moreover, the *"simulation scenario"* and *"record year"* are two attributes frequently used in the where clause of the queries referring these analytical objects. Regarding the Cassandra column family *"smart meters records"*, it is possible to use *"sm id"* as the partition key, balancing the data throughout the nodes in the cluster according to the identifier of the smart meter. In Cassandra, the partition key is the first part of the primary key, which in our case is a compound key using *"sm id"* and *"record date"*. In Hive, we do not need to define primary keys.

As can be seen in Fig. 2, analytical objects can be joined together to answer certain queries. These join operations are optional, due to the fact that analytical objects can be modeled without any external references to other objects, which is the case for most analytical objects in the SusCity BDW. There is no need for declaring foreign keys at object creation and analytical objects can be joined using all attributes whose values match. These join operations are severely different from the join operations required between fact tables and dimension tables, since we only use them in queries that relate different analytical objects, which is much less frequent than joining fact tables and dimensions for each query. Considering the SusCity BDW, the *"building id"* attribute is present in three analytical objects: *"buildings general indicators"*; *"buildings energy consumption"*; and *"smart meters load"*. Instead of replicating the information about buildings in these three objects and creating unnecessary redundancy, taking into consideration that the *"buildings general indicators"* object is relatively small (around 60,000 rows for the city of Lisbon), it can be easily joined with the other two tables, in order to answer specific questions. Nevertheless, all three objects are completely independent and are capable of answering different queries without relying on any join operations.

Complementing what was already mentioned regarding querying and OLAP, it is possible to use Presto to perform joins and unions between batch analytical objects (Hive tables) and streaming analytical objects (Cassandra column families), providing useful insights extracted from historical and real-time data, as can be seen in Fig. 2. The size of the datasets being joined is of major relevance for an adequate query performance, and should be severely taken into consideration. This is the reason why analytical objects should almost never be joined in their raw format. First, we need to aggregate and filter (as much as possible) each analytical object involved in the join operation. The larger the inputs in each side of the join operation, the complex and slower the query becomes. In this case, materialized views stored in Hive tables are significantly helpful for maintaining interactive response times in query execution and the responsiveness of the data visualization platform.

Summarizing the main strategies for this modelling approach, we can highlight three major strategies:

- Use fully denormalized structures to avoid the cost of join operations in Big Data environments;
- The use of nested structures, which are not typically found in traditional modelling techniques, can provide more flexibility and performance advantages in several scenarios;
- Divide data flows and storage into batch and streaming, as discussed and explored in [37], but that does not to imply different modeling strategies for each data velocity. This analytical objects approach can be used for both batch and streaming.

## 4.2 Data Transfer Between the Batch and Streaming Storage Systems

Finally, the last topic needing discussion regarding the SusCity data modeling strategy is the data transfer from the streaming storage to the batch storage. The need to transfer the data was already highlighted in section 3, but it will be quantitatively evaluated in this section. As mentioned, NoSQL databases are OLTP-oriented [3], unlike Hive which is an OLAP-oriented technology. Typically, OLTP systems relax sequential access efficiency for random access efficiency. Therefore, systems like Cassandra are adequate for the multiple random write operations demanded by the real-time collection of data from hundreds or thousands of smart meters. However, these systems lack the efficiency to process (e.g., aggregate) a large amount of historical data, which is frequently demanded by OLAP queries. Table 1presents the results from an experimental evaluation

conducted in the infrastructure and testbed of the SusCity research project, evaluating the response times when submitting Presto queries to Hive tables and Cassandra column families.

As demonstrated in Table 1, given the same analytical object and the same amount of data, Presto OLAP queries on Hive Tables (ORC file format) perform significantly faster than the queries on Cassandra column families. This corroborates the statements previously stated and the decision of periodically move historical data from Cassandra to Hive, maintaining only the most recent data in Cassandra. The periodicity of this data transfer depends on the specific requirements regarding interactivity in response times, the volume of data being stored and the available infrastructure. Data can be transferred on a daily, weekly or monthly basis, for example.

**Table 1: Performance comparison between analytical objects stored in Hive and Cassandra.**

| Query | Input Rows | Output Rows | Hive | Cassandra |
|---|---|---|---|---|
| Show the last 10 records from all smart meters. | ~2.8 million | 10 | 0.56s | 3.08s |
| Calculate the average of kWh recorded by each smart meter. | ~2.8 million | 214 | 0.56s | 4.2s |
| Count how many records a certain smart meter contains. | ~2.8 million | 1 | 0.74s | 0.98s |

## 5 The SusCity Data Visualization Platform

Throughout this paper, we focused on the logical and physical layers of the BDW. In this section, we highlight some relevant use cases in which the data visualization platform can help the city's stakeholders in the decision-making process. As previously presented, the SusCity data visualization platform was developed using modern JavaScript libraries like the Google Maps API V3 and Chart.js. Obviously, since it is a web-based platform, core languages are also present (HTML, CSS and pure JavaScript), as well as other supporting JavaScript libraries like JQuery [27]. It is a platform purely based on a service-oriented architecture, using Java REST web services to establish the communication between the JavaScript components of the platform and the querying and OLAP engine instantiated by Presto. Each query to the BDW goes to this REST backend for an adequate modularity of the platform. Using this service-oriented and modular approach, it becomes easier to update or replace components and technologies if that need arises in the future.

In this section, we will briefly present several dashboards under development in the SusCity research project (Fig. 3 and Fig. 4), which can also be interesting for other Smart Cities initiatives. Due to security and privacy issues, values have been omitted and locations have been changed/deleted, not referencing any sensitive information regarding specific areas or buildings in the city. The first dashboard (Fig. 3) is based on the energy consumption of each parish in the city (2 parishes in the SusCity testbed). Decision-makers are able to understand the energy consumption in each parish, analyzing the city's consumption by hour, time period (e.g., morning or afternoon) or by quarter. Users can interact with multiple parishes by clicking on them, revealing specific energy consumption for specific parishes, and comparing it with the overall consumption of the city, with the goal of extracting insights regarding critical zones in the city, for example.

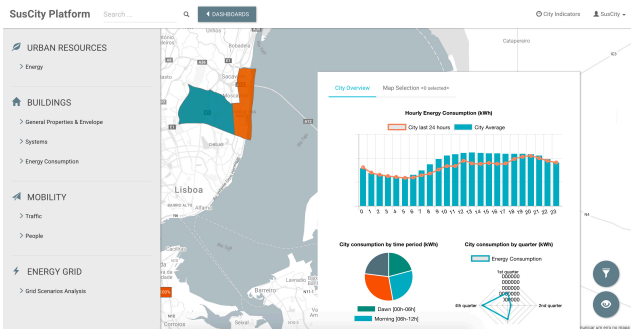


**Figure 3: Energy consumption dashboard in the SusCity data visualization platform.**

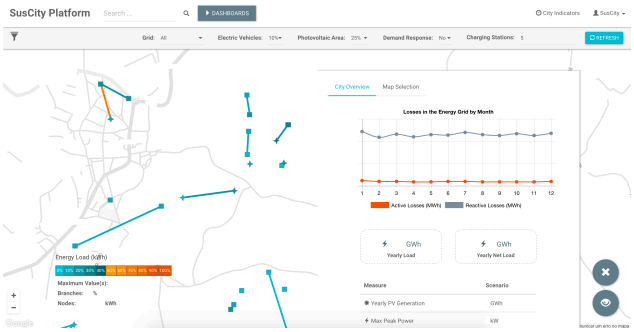


**Figure 4: Dashboard for energy grid simulation in the SusCity data visualization platform.**

As can be seen in the left side of Fig. 3, several dashboards can be deployed in the data visualization platform, focusing on several areas of intervention in the city. Other dashboards focus on the analysis of the buildings in the city, including information about their cooling/heating systems, energy consumption, energy class, and envelope properties (e.g., window glass type and window materials). In the SusCity data visualization platform, we can also make available the predictive capabilities of the SusCity BDW, such as the segmentation (clustering) of buildings according to their energy consumption, and the respective energy forecasting for the next days or weeks, as briefly mentioned in this paper and as also presented in [11].

Finally, a dashboard to simulate and analyze stress scenarios in the energy grid can be significantly useful in the context of

Smart Cities, as depicted in Fig. 4. Each scenario corresponds to a set of input parameters (e.g., number of electrical vehicles, photovoltaic area, number of charging stations) that may affect the behavior of the energy grid, such as energy losses, load, and maximum peak power. In the SusCity BDW, the results of the simulations for these scenarios are stored in analytical objects, as presented in the previous section, and the data visualization platform can use the querying and OLAP engine to extract and provide useful insights for stakeholders interested in the impact on the energy grid caused by certain initiatives. Due to the modular and service-oriented nature of the SusCity data visualization platform, and the flexible and scalable SusCity BDW, we are able to provide dashboards for decision-makers, regardless of data volume, variety and velocity, without being hold back by rigid data modelling techniques, and complex data collection, preparation and enrichment pipelines.

## 6 CONCLUSIONS

This paper proposed a novel and flexible approach for Big Data Warehousing in the context of Smart Cities. Given the need for scientific advancements in the topic of Big Data Warehousing and Smart Cities, this paper presented a set of foundations for designing and implementing BDWs: the logical components and data flows of the BDW system; the technologies used to instantiate the logical components; the infrastructure in which the technologies can be deployed; and the data modelling strategy for flexible and scalable BDWs. The approach proposed in this paper also focuses on the collection, preparation and enrichment of data arriving in batches and through streaming mechanisms, including the output of data mining algorithms and simulation models, when useful for certain use cases in Smart Cities. The SusCity Big Data Warehousing approach has been effectively and efficiently supporting the SusCity [52] platform in the context of Smart Cities, making available several analytical dashboards and simulation mechanisms, supporting the decision-making process of several stakeholders in the city of Lisbon.

This approach has been effective and efficient not only in the context of Smart Cities, but also in other proof-of-concepts under implementation. It represents a departure from the traditional relational DWs, making extensive use of emergent Big Data techniques and technologies, such as Hadoop, Spark and NoSQL databases. Practitioners can follow this approach to build their own BDWs in several organizations spread across different business areas. With this work, we intend to fill in some scientific gaps and foster future research in Big Data Warehousing, which is a topic lacking from contributions regarding models and methods for the design and implementation of BDWs, properly evaluated through benchmarking or proof-of-concept implementation.

For future work, we aim to enrich and generalize the approach proposed in this paper, in order to provide a more general approach for Big Data Warehousing. This will be possible through the development of general models and methods, validated through Big Data benchmarks and more proof-of-concept implementations in different business areas.

## 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 the SusCity project, MITP-TB/CS/0026/2013.

## REFERENCES

[1] Anderson, J.C. et al. 2010. *CouchDB: the definitive guide*. O'Reilly Media, Inc.
[2] Apache Kafka Homepage: 2017. *https://kafka.apache.org/*. Accessed: 2017-03-05.
[3] Cattell, R. 2011. Scalable SQL and NoSQL data stores. *ACM SIGMOD Record*. 39, 4 (2011), 12–27.
[4] Chart.js | Open source HTML5 Charts: 2017. *http://www.chartjs.org/*. Accessed: 2017-03-05.
[5] Cheng, B. et al. 2015. Building a big data platform for smart cities: Experience and lessons from santander. *Big Data (BigData Congress), 2015 IEEE International Congress on* (2015), 592–599.
[6] Chevalier, M. et al. 2017. Document-oriented Models for Data Warehouses - NoSQL Document-oriented for Data Warehouses. (Mar. 2017), 142–149.
[7] Chevalier, M. et al. 2015. Implementing multidimensional data warehouses into NoSQL. *International Conference on Enterprise Information Systems (ICEIS 2015)* (2015), 172–183.
[8] Clegg, D. 2015. Evolving data warehouse and BI architectures: The big data challenge. *TDWI Business Intelligence Journal*. 20, 1 (2015), 19–24.
[9] Costa, C. and Santos, M.Y. 2017. A Conceptual Model for the Professional Profile of a Data Scientist. (Apr. 2017).
[10] Costa, C. and Santos, M.Y. 2016. BASIS: A big data architecture for smart cities. *2016 SAI Computing Conference (SAI)* (Jul. 2016), 1247–1256.
[11] Costa, C. and Santos, M.Y. 2015. Improving cities sustainability through the use of data mining in a context of big city data. *The 2015 International Conference of Data Mining and Knowledge Engineering* (2015), 320–325.
[12] Costa, C. and Santos, M.Y. 2016. Reinventing the Energy Bill in Smart Cities with NoSQL Technologies. *Transactions on Engineering Technologies*. S. Ao et al., eds. Springer Singapore. 383–396.
[13] Dean, J. and Ghemawat, S. 2008. MapReduce: Simplified Data Processing on Large Clusters. *Commun. ACM*. 51, 1 (Jan. 2008), 107–113.
[14] Floratou, A. et al. 2014. SQL-on-Hadoop: Full Circle Back to Shared-nothing Database Architectures. *Proc. VLDB Endow.* 7, 12 (Aug. 2014), 1295–1306.
[15] Foo, A. 2013. Is the data warehouse dead? *IBM Data Management Magazine*. 5 (2013).
[16] Girtelschmid, S. et al. 2013. Big data in large scale intelligent smart city installations. *Proceedings of International Conference on Information Integration and Web-based Applications & Services* (2013), 428.
[17] Golab, L. and Johnson, T. 2014. Data stream warehousing. *2014 IEEE 30th International Conference on Data Engineering (ICDE)* (Mar. 2014), 1290–1293.
[18] Google Maps JavaScript API: 2017. *https://developers.google.com/maps/documentation/javascript/*. Accessed: 2017-03-05.
[19] Goss, R.G. and Veeramuthu, K. 2013. Heading towards big data building a better data warehouse for more data, more speed, and more users. *Advanced Semiconductor Manufacturing Conference (ASMC), 2013 24th Annual SEMI* (2013), 220–225.
[20] Gröger, C. et al. 2014. The Deep Data Warehouse: Link-Based Integration and Enrichment of Warehouse Data and Unstructured Content. *IEEE 18th International Enterprise Distributed Object Computing Conference (EDOC)* (Sep. 2014), 210–217.
[21] Hall, M. et al. 2009. The WEKA data mining software: an update. *ACM SIGKDD explorations newsletter*. 11, 1 (2009), 10–18.

[22] Hevner, A.R. et al. 2004. Design Science in Information Systems Research. *MIS Q.* 28, 1 (Mar. 2004), 75–105.

[23] Hive Transactions - Apache Hive - Apache Software Foundation: 2017. *https://cwiki.apache.org/confluence/display/Hive/Hive+Transactions*. Accessed: 2017-01-30.

[24] Hortonworks 2016. *Solving Apache Hadoop Security: A Holistic Approach to a Secure Data Lake*. Hortonworks.

[25] Huai, Y. et al. 2014. Major Technical Advancements in Apache Hive. *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data* (New York, NY, USA, 2014), 1235–1246.

[26] Jara, A.J. et al. 2013. Determining human dynamics through the internet of things. *Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)-Volume 03* (2013), 109–113.

[27] jQuery: 2017. *https://jquery.com/*. Accessed: 2017-03-05.

[28] Kearney, M. 2012. Embracing big data from the warehouse. *IBM Data Management Magazine*.

[29] Khan, Z. et al. 2013. Cloud based big data analytics for smart future cities. *Proceedings of the 2013 IEEE/ACM 6th international conference on utility and cloud computing* (2013), 381–386.

[30] Kimball, R. and Ross, M. 2013. *The data warehouse toolkit: The definitive guide to dimensional modeling*. John Wiley & Sons.

[31] Kobielus, J. 2012. Hadoop: Nucleus of the next-generation big data warehouse. *IBM Data Management Magazine*.

[32] Kornacker, M. et al. 2015. Impala: A modern, open-source sql engine for hadoop. *Proc. CIDR'15* (California, USA, 2015).

[33] Krishnan, K. 2013. *Data Warehousing in the Age of Big Data*. Morgan Kaufmann Publishers Inc.

[34] Li, X. and Mao, Y. 2015. Real-Time data ETL framework for big real-time data analysis. *2015 IEEE International Conference on Information and Automation* (Aug. 2015), 1289–1294.

[35] Lipcon, T. et al. 2015. *Kudu: Storage for Fast Analytics on Fast Data*. Cloudera.

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

[37] Marz, N. and Warren, J. 2015. *Big Data: Principles and best practices of scalable realtime data systems*. Manning Publications Co.

[38] Mohanty, S. et al. 2013. *Big Data imperatives: enterprise Big Data warehouse, BI implementations and analytics*. Apress.

[39] NBD-PWG 2015. *NIST Big Data Interoperability Framework: Volume 6, Reference Architecture*. Technical Report #NIST SP 1500-6. National Institute of Standards and Technology.

[40] Peffers, K. et al. 2007. A Design Science Research Methodology for Information Systems Research. *J. Manage. Inf. Syst.* 24, 3 (Dec. 2007), 45–77.

[41] Philip Chen, C.L. and Zhang, C.-Y. 2014. Data-intensive applications, challenges, techniques and technologies: A survey on Big Data. *Information Sciences*. 275, (Aug. 2014), 314–347.

[42] Presto | Distributed SQL Query Engine for Big Data: 2016. *https://prestodb.io/*. Accessed: 2016-10-23.

[43] Qiao, L. et al. 2015. Gobblin: Unifying data ingestion for Hadoop. *Proceedings of the VLDB Endowment*. 8, 12 (2015), 1764–1769.

[44] Russom, P. 2016. *Data Warehouse Modernization in the Age of Big Data Analytics*. The Data Warehouse Institute.

[45] Russom, P. 2014. *Evolving Data Warehouse Architectures in the Age of Big Data*. The Data Warehouse Institute.

[46] Santos, M.Y. and Costa, C. 2016. Data Models in NoSQL Databases for Big Data Contexts. *2016 International Conference of Data Mining and Big Data (DMBD)* (2016), 1–11.

[47] Santos, M.Y. and Costa, C. 2016. Data Warehousing in Big Data: From Multidimensional to Tabular Data Models. *Ninth International C* Conference on Computer Science & Software Engineering (C3S2E)* (2016), 51–60.

[48] Shanahan, J.G. and Dai, L. 2015. Large scale distributed data science using apache spark. *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2015), 2323–2324.

[49] Shvachko, K. et al. 2010. The Hadoop Distributed File System. *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)* (May 2010), 1–10.

[50] Simmhan, Y. et al. 2013. Cloud-based software platform for big data analytics in smart grids. *Computing in Science & Engineering*. 15, 4 (2013), 38–47.

[51] Song, J. et al. 2015. HaoLap: A Hadoop based OLAP system for big data. *Journal of Systems and Software*. 102, (Apr. 2015), 167–181.

[52] SUSCITY – An MIT Portugal Project: 2016. *http://suscity-project.eu/inicio/*. Accessed: 2016-05-04.

[53] Talend Open Studio for Big Data Product Details: 2017. *https://www.talend.com/download_page_type/talend-open-studio/*. Accessed: 2017-03-05.

[54] Thusoo, A. et al. 2010. Data Warehousing and Analytics Infrastructure at Facebook. *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data* (New York, NY, USA, 2010), 1013–1020.

[55] Thusoo, A. et al. 2010. Hive-a petabyte scale data warehouse using hadoop. *IEEE 26th International Conference on Data Engineering (ICDE)* (2010), 996–1005.

[56] Vilajosana, I. et al. 2013. Bootstrapping smart cities through a self-sustainable model based on big data flows. *IEEE Communications magazine*. 51, 6 (2013), 128–134.

[57] Wang, H. et al. 2015. Efficient query processing framework for big data warehouse: an almost join-free approach. *Frontiers of Computer Science*. 9, 2 (2015), 224–236.

[58] Wang, H. et al. 2011. LinearDB: A Relational Approach to Make Data Warehouse Scale Like MapReduce. *Database Systems for Advanced Applications*. J.X. Yu et al., eds. Springer Berlin Heidelberg. 306–320.

[59] Wang, S. et al. 2014. High dimensional biological data retrieval optimization with NoSQL technology. *BMC Genomics*. 15 Suppl 8, (2014), S3–S3.

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

[61] Welcome to Apache Hadoop: 2016. *https://hadoop.apache.org/*. Accessed: 2017-02-01.

[62] Zikopoulos, P. and Eaton, C. 2011. *Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data*. McGraw-Hill Osborne Media.