Hadoop based Demography Big Data Management System

Syeda Sana Bukhari
College of Information and Communication Engineering
Sungkyunkwan University (SKKU)
Suwon, 16419, South Korea
email: sanabukhari1992@gmail.com

JinHyuck Park
College of Information and Communication Engineering
Sungkyunkwan University (SKKU)
Suwon, 16419, South Korea
email: vkqxkr@gmail.com

Dong Ryeol Shin
College of Information and Communication Engineering
Sungkyunkwan University (SKKU)
Suwon, 16419, South Korea
email: drshin@skku.edu

***Abstract*—Demography relates to statistical measures in social sciences, which is the study of human population in a certain period. Its main focus is to calculate birth & death rate, mobility, literacy ratio, and marriages. The immense population growth is presenting new challenges to handle massive quantities of demographic data (big data). With the evolution of computer age and software development, different procedures have been introduced to deal with big data. Most of the systems are based on traditional rational database, which lack in their capability to handle big data efficiently and accurately. Apache Hadoop platform is becoming famous for handling big data problems. Herein, we propose a demography big data handling system under Hadoop ecosystem to solve the issue of demography large-scale data handling. The main components of our system are Sqoop, HBase and Hive. Our system defines a method to export data from different RDBMS to Hadoop HBase. Moreover, HBase-Hive integration is a good architecture to store and query data for analysis efficiently. The results indicate that the presented approach better handles the distributed datasets as a single dataset to analyze for future estimates. Our approach will provide advancement in handling sophisticated demography big data and will eventually improve analytics such as promoting sustainable agriculture, preventing poverty, and decreasing death rate.**

Keywords—RDBMS, HBase, Hive, Big Data Analysis, MapReduce, Demography Big Data.

I. INTRODUCTION

Demography has been compelled by data since its commencement. As it's the era of computer sciences, social science field is merging with computer field to leverage the capacity to collect and analyze data at the high scale that may reveal results of individual and group behavior. Computational social science is the new area that is providing solutions to deal with demography. The size of demographic data has extended as the population is growing; hence, the number of schools, colleges, and hospitals is also increasing having their own management systems based on a traditional database for record keeping. With the advancement of social media, the literacy rate is also increasing in almost all countries especially in the developed ones. There are various sources of demography records even in a single country. For annual demography records storing and analysis, we need to collect data from these different resources. Currently used data gathering schemes are collecting samples, gathering registered data records and census data etc. The larger datasets or samples used for analysis always help in more accurate estimates. Demography data is covering the big data 3 V's concerns as shown in Fig. 1.

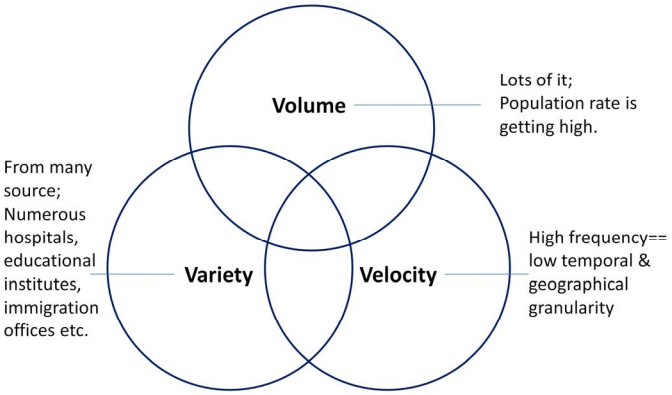

Fig. 1. Demography Big Data

Dealing with big data records related to any area is becoming a challenge since history is also essential to maintain for future analysis. Only storing demography data records efficiently with a fault-tolerant system is not sufficient, there is a need to use these big data records for several decisions making, analysis, important calculations like mortality and ratio calculation of literacy, and mobility etc. Demography Big Data analytics can be used to prevent poverty, decrease death rate and for many other important aspects as well.

To address this challenge we are introducing Hadoop [1] based demographic big data handling system. Our system is capable to merge scattered DBMS records in Apache HBase storage as a single record. It also facilitates to add sampled survey data with those records in HBase manually. Sqoop [2] tool is used to import data from different RDBMS to HBase [3] and Apache Hive [4] to query results for analysis.

II. BACKGROUND

In this section, we give a brief overview of used Hadoop ecosystem parts to build our demographic big data management system.

978-1-5386-5889-5/18/$31.00 ©2018 IEEE
SNPD 2018, June 27-29, 2018, Busan, Korea

A. Hadoop & MapReduce

Hadoop is an open source, scalable and reliable architecture created by Dong Cutting in 2004. In January 2008, Hadoop became a top-level Apache Software Foundation project. It is used for large-scale distributed unstructured data storage. Hadoop has many components that have their own special functionalities as shown in the following table.

TABLE I. HADOOP ECO SYSTEM

Hadoop Eco System			
Characteristics			
Data Storage	***Data Processing***	***Data Access***	***Data Management***
HDFS (Hadoop distributed file system)	MapReduce (Cluster Management)	Hive (HiveQL)	Zookeeper (Management)
		Pig (Dataflow)	Flume (Monitoring)
HBase (Column DB storage)	YARN (Cluster & resource management)	Mahout (Machine Learning)	Chukwa (Monitoring)
		Sqoop (RDBMS connection)	Oozie (workflow monitoring)
		Avro RPC serialization	

MapReduce framework is for distributed computing. It splits a large task into multiple jobs and runs them in parallel by mapping key-value pairs [5]. Job Tracker node is responsible to divide a large job into MapReduce subtasks and then gives these to Task Tracker nodes. MapReduce scheme provides high performance for distributed data handling [1].

B. Sqoop

Sqoop [6] is a tool of Hadoop ecosystem that is used to create a connection between Hadoop and RDBMS. It provides a facility to transfer data between rational databases such as Oracle, MySQL and Hadoop HDFS, HBase or Hive.

C. HBase

HBase is a column based, large-scale, open-source, distributed database management tool [7]. It has a master and multiple region servers. The master server is responsible for assigning jobs to region server; it uses Zookeeper service to coordinate with region servers. HBase uses Zookeeper to track distributed data status. HBase has been used for large-scale data storage in many projects.

D. Hive

Hive is a big data warehouse system that uses SQL like query language HiveQL to query data [4]. Hive has a Metastore that contains statistics and schemas. It also includes execution engine that processes the query and produces results. It was introduced by Facebook and is currently used for their messages service [8]. Hive driver manages session and lifecycle of a HiveQL statement. It is capable of storing Terabytes of data and is being used for both ad-hoc analysis and reporting.

III. RELATED WORK

SQL is not supported by HBase. In order to fetch results, there is a need to write MapReduce jobs. However, there are some Hive/HBase integration projects that allow SQL statements be written in Hive to access HBase tables. Apache Software Foundation is also providing support for the integration of Hive and HBase that allows HiveQL statements to access HBase tables for both read and write [9]. Vora [7] used HBase to perform random reads/writes on very large datasets containing image files and the results were demonstrated to be better than using MySQL on such datasets. Facebook is a large project that is also using Hadoop based solutions for its big data. Facebook is using Hive in messages service and Hbase for long-term storage of chat messages [10] because it is elastic, cheap, fault tolerant and has high write throughput. Hadoop-Based Healthcare Information [11] used Sqoop tool to import/export data between MySQL database and HDFS, and it was shown that Hadoop-based healthcare architecture is highly effective to build a healthcare information system. Gao et al. [13] presented a comparison of traditional RDBMS, with their HBase storage model and proved Hbase is a better solution for the storage of geospatial data.

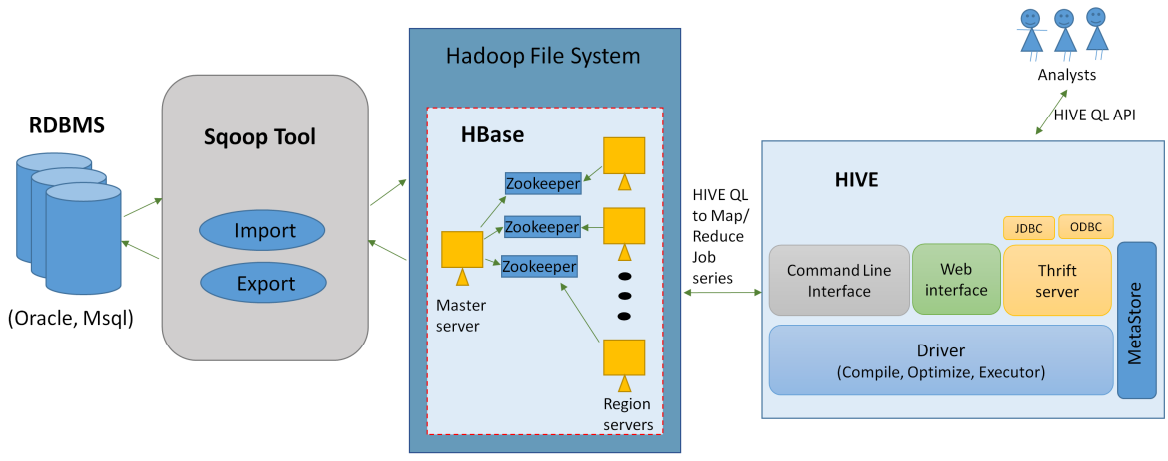

Fig. 2. Demography Big Data Management System Architecture

IV. Proposed System

Fig. 2 illustrates our proposed architecture. We used Apache Hive to query results for analysis. When any query is received by hive driver, it proceeds the query. The compiler makes a plan by getting Metadata for request query. The execution engine decodes the HiveQL request to a series of MapReduce jobs with storage handler and those jobs are sent to HBase. HBase executes those jobs and replies to Hive Engine with desired results after successful job completion. The storage handler interface performs the main role to link HBase and Hive.

Our proposed Hadoop based demography big data management system can be divided into two parts; 1) storage of demographic big data as a single unit by importing from different RDBMS, 2) data retrieval for analysis and estimations. Following is the detailed explanation of these two divisions.

A. Storage

The first half is to import records of scattered DBMS to HBase. Traditional databases import slowly into native representation. It will be time taking to store all Demography big data in a large RDBMS from small databases. Databases use a structured mechanism of storage and these are lacking in their capacity to handle the huge amount of data [12]. Hadoop platform does scale to large amounts of data more efficiently. We know that it's possible to make tables to store data using Hive and later query it using HiveQL.

However, in our system, we use HBase to store data. The main reason to import data to HBase, and not in Hive, is that Hive at its core is a query engine whereas HBase is built for data storage. It is difficult to import data directly into Hive from RDBMS using Sqoop. On another side, Sqoop directly imports data into HBase instead of HDFS directory [6]. Using Sqoop our system imports table records from different databases. Following listing 1 shows how we import the whole table at once using Sqoop to HBase; this saves time for creating a table and inserting record by record. In case the table does not exist, it will automatically create the table with the same name before importing data.

Listing 1. Importing database table records into Hbase table

```
sqoop import
--connect jdbc:mysql:// localhost/hospital
--username root
--password password
--table DeathRecord
--hbase-table deathRecordData
--column-family deathRecordId
--hbase-row-key id
–m 1
```

Fig. 3 is showing the workflow of our system's first half. At first, Sqoop builds a connection with the database, and after successful connection, it proceeds with the request for table. Database engine always responds to fetching requests sent by Sqoop. It gets Metadata from Metastore and fetches table from Datastore. Eventually, data is imported into Sqoop, which exports table data to HBase. Most actions are hidden from the user. Sqoop also uses MapReduce scheme to complete jobs in earlier time.

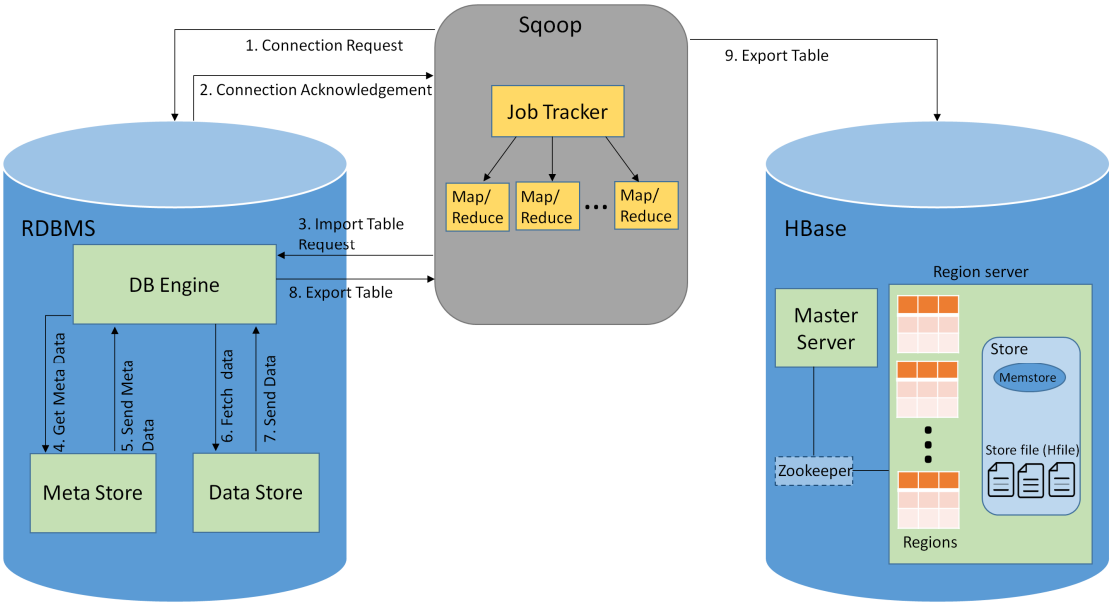

Fig. 3. RDBMS to HBase table data transfer using Sqoop

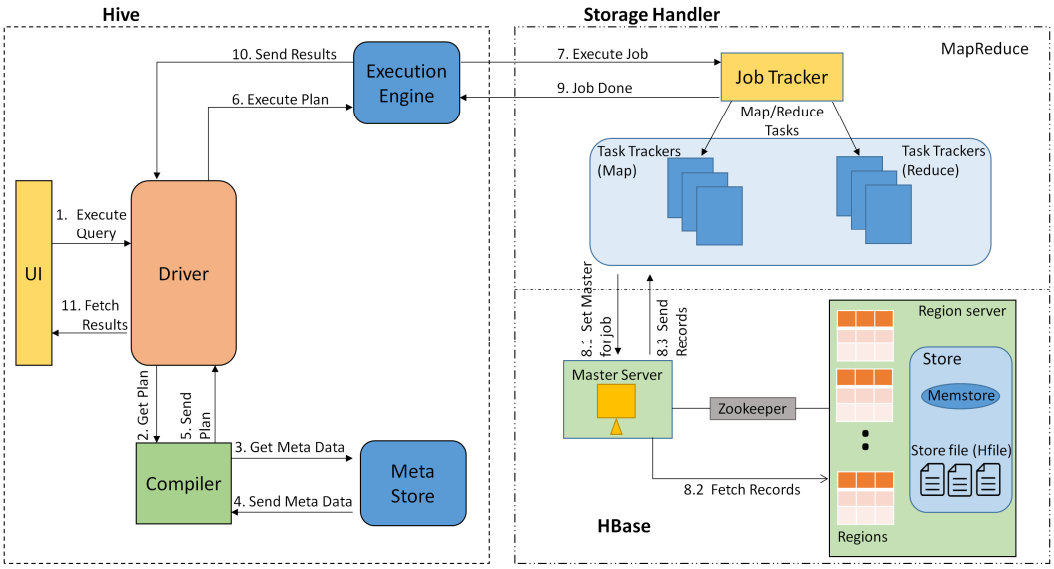

Fig. 4. Hive and HBase integration workflow

B. *Data Retrieval and Analysis*

Hive has some restraints of high latency however, HBase lacks analytical capabilities. By integrating these two technologies will produce the best possible solution. The second half of proposed system has basic functionality to interact with Demography big data stored in HBase from Hive. When any query for analysis or estimation is received by Hive driver from a user interface, its compiler approaches Metastore and receives Metadata of query. Afterwards, Hive execution engine starts executing the query as shown in Fig. 4. Apache provides Storage Handler [9] that does the basic function of the interaction between Hive and HBase.

Listing 2. HiveQL queries

Display Records from deathRecordTable where age lays between 16 to 30 and death cause is "heart attack"

```
hive> SELECT * FROM deathRecordData WHERE
      age> 16 AND age < 30 AND deathReason ==
      "heart attack" ;
```

Display ratio of death caused by "heart attack"

```
hive > SELECT SUM(IF (deathRecordData.deathReason
      == 'heart attack', 1 , 0))/COUNT(*) * 100 as
      heartAttackRatio
      FROM deathRecordData
      GROUP BY deathRecordData.deathReason;
```

Count the occurrences of each different "deathReason" values

```
hive> SELECT deathReason, count(*) as deathReason
      FROM deathRecordData
      GROUP BY deathReason ORDER BY
      deathReason;
```

Queries have been written in HiveQL as shown in Listing 2. Storage Handler allows us to access data stored in HBase, therefore, we have directly used the name of HBase table in the query.

As shown in Fig. 4, storage handler is responsible for converting the requested query into series of jobs. Storage handler associates suitable master server for each job. Apache zookeeper helps master server to interact with region server and maintains a track of their actions. Demography data is stored in the form of regions (tables) in HBase. The master server manages region servers and their regions. Region server approaches the desired region for result calculation by using Metadata table that have been sent with a query. After the Master server has the results, it sends them back to storage handler that forwards it to Hive. In the last step, Hive displays results on the user interface. Hive fulfills ACID property of transactions with maximum productivity.

We imported census dataset to MySQL database that contains information about deaths rates for the 10 leading causes of death in the United States from 1999 to 2015. Later on we use our system that imported tables in HBase using Sqoop. Afterwards, we used the dataset in HBase-Hive integrated system for analyzing average death rate in United States according to years for two leading death causing diseases 'Cancer' and 'Diseases of Heart'. Fig. 5 shows two leading causes of death analysis.

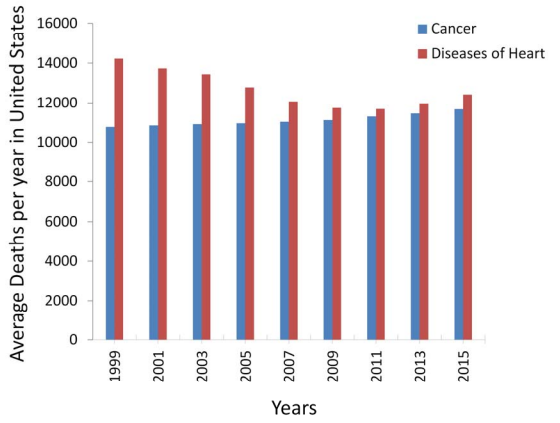

Fig. 5. Average deaths per year in US for two leading causes

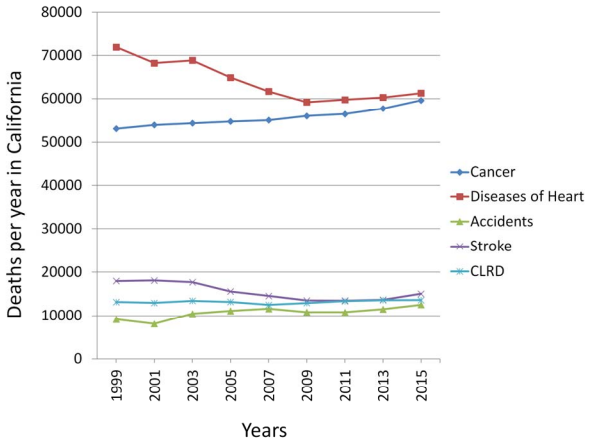

Fig. 6. Deaths per year in California for five leading causes

Secondly, we analyze which state has the highest death rate in the United States. For both 'Cancer' and 'Diseases of Heart' the highest death rate narrated by analysis was in the state of California shown in table below.

TABLE II. HIGHEST DEATH RATE STATE FOR 'CANCER' AND 'DISEASES OF HEART'

Year	Cause Name	State	Deaths
1999	Cancer	California	**53067**
	Diseases of heart		**71930**
2001	Cancer	California	**53924**
	Diseases of heart		**68234**
2003	Cancer	California	**54319**
	Diseases of heart		**68864**
2005	Cancer	California	**54732**
	Diseases of heart		**64916**
2007	Cancer	California	**55011**
	Diseases of heart		**61690**
2009	Cancer	California	**55991**
	Diseases of heart		**59206**
2011	Cancer	California	**56449**
	Diseases of heart		**59772**
2013	Cancer	California	**57714**
	Diseases of heart		**60299**
2015	Cancer	California	**59629**
	Diseases of heart		**61289**

Furthermore, we analyze five leading death causes that have highest death rate for California during the time interval of 1999 to 2015.

Analysis in Fig. 6 reveals that diseases of heart are main cause of death in California as well as for other states shown in Fig. 5. The death rate for 'Cancer' is increasing with the passage of time. Accidental death rate is also increasing although it is negligible in front of 'diseases of heart' death rate.

V. DISCUSSION

The purpose of this study was to evaluate demography as big data as it is increasing immensely over time. RDBMS are not flexible enough to deal with highly increasing data, however, HBase is capable for handling growing data with high performance and less cost. Table III is showing why HBase storage is better than traditional database storage.

TABLE III. RATIONAL DATABASE VS HBASE

Rational Database	HBase
Supports scale up: If more disk space and memory processing power is required, upgrade system with more powerful server.	**Supports scale out:** If more disk space and memory processing power is required, do not upgrade the server. Instead, add new servers to the cluster.
SQL queries are used to read stored data table.	Need to write MapReduce programs to access tables.
It has fixed schema.	It has flexible schema.
Only supports structured data.	Supports both semi-structured and unstructured data.

HBase is not user friendly interface as it does not support easy querying. Hence, we used HBase-Hive integration to improve retrieval performance. We queried through HiveQL that is similar to SQL, which gives high throughput than SQL query. HBase-Hive outperforms MySQL in large datasets size of terabyte and more. We proposed Hbase-Hive architecture to store demography data as it lies in big data category by extensively increasing rate. Its records are always collected from different sources. Thereby, the proposed system can merge different sources data without caring for the schema of records and consuming less time.

VI. Conclusion and Future Work

Hadoop ecosystem components have overcome many DBMS which were unable to deal with big data. Demography data storage and analysis system is able to deal with large datasets efficiently. Apache Hadoop is getting famous these days for giving solutions to manage big data. Hence, we also determined big data handling approach for demographic data since it falls under big data. Firstly, we defined the components that were playing a role to store and used demography big data for analysis. Secondly, we gave an explanation of our system's architecture and the workflow of its two respective divisions. Data collection from different sources was done with relative ease and greater efficiency. Rather than using sample data for analyzing demographics, the suggested approach used large data as an input to investigate useful information for variant estimations.

Our approach is based on solving the storage problem of demography big data, by gathering it from different sources, furthermore, query it for estimations and decision making. In future, we will use Spark with Spark SQL for querying, and re-evaluating results for improving the overall system performance. Machine learning algorithms will also be added for more analysis in current system.

Acknowledgment

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2017R1D1A1B03032855).

References

[1] Apache Hadoop. Available at http://wiki.apache.org/hadoop.

[2] Sqoop. https://sqoop.apache.org/

[3] HBase. https://hbase.apache.org/

[4] Hive. Apache Software Foundation, 2012. https://hive.apache.org/

[5] Jeffrey Dean , Sanjay Ghemawat, MapReduce: simplified data processing on large clusters, Communications of the ACM, v.51 n.1, January 2008[doi>10.1145/1327452.1327492]

[6] Jain, Ankit. Instant Apache Sqoop. Packt Publishing Ltd, 2013.

[7] Mehul Nalin Vora, "Hadoop-HBase for large-scale data," Proceedings of 2011 International Conference on Computer Science and Network Technology, Harbin, 2011, pp. 601-605. doi: 10.1109/ICCSNT.2011.6182030.

[8] A. Thusoo et al., "Hive - a petabyte scale data warehouse using Hadoop," 2010 IEEE 26th International Conference on Data Engineering (ICDE 2010), Long Beach, CA, 2010, pp. 996-1005. doi: 10.1109/ICDE.2010.5447738

[9] https://cwiki.apache.org/confluence/display/Hive/HBaseIntegration

[10] Aiyer, A.S., Bautin, M., Chen, G.J., Damania, P., Khemani, P., Muthukkaruppan, K., Ranganathan, K., Spiegelberg, N., Tang, L., Vaidya, M. "Storage infrastructure behind Facebook Messages: using HBase at scale". IEEE Data Eng. Bull. 35(2), 413 (2012)

[11] Hongsong Chen and Zhongchuan Fu, "Hadoop-Based Healthcare Information System Design and Wireless Security Communication Implementation," Mobile Information Systems, vol. 2015, Article ID 852173, 9 pages, 2015. doi:10.1155/2015/852173

[12] S. Madden, "From Databases to Big Data," in IEEE Internet Computing, vol. 16, no. 3, pp. 4-6, May-June 2012. doi: 10.1109/MIC.2012.50

[13] F. Gao, P. Yue, Z. Wu and M. Zhang, "Geospatial data storage based on HBase and MapReduce," 2017 6th International Conference on Agro-Geoinformatics, Fairfax, VA, 2017, pp. 1-4. doi: 10.1109/Agro-Geoinformatics.2017.80470