Designing and Implementing Data Warehouse for Agricultural Big Data

Vuong M. Ngo(✉), Nhien-An Le-Khac, and M-Tahar Kechadi

School of Computer Science, University College Dublin, Dublin, Ireland
{vuong.ngo,an.lekhac,tahar.kechadi}@ucd.ie

Abstract. In recent years, precision agriculture that uses modern information and communication technologies is becoming very popular. Raw and semi-processed agricultural data are usually collected through various sources, such as: Internet of Thing (IoT), sensors, satellites, weather stations, robots, farm equipment, farmers and agribusinesses, etc. Besides, agricultural datasets are very large, complex, unstructured, heterogeneous, non-standardized, and inconsistent. Hence, the agricultural data mining is considered as Big Data application in terms of volume, variety, velocity and veracity. It is a key foundation to establishing a crop intelligence platform, which will enable resource efficient agronomy decision making and recommendations. In this paper, we designed and implemented a continental level agricultural data warehouse by combining Hive, MongoDB and Cassandra. Our data warehouse capabilities: (1) flexible schema; (2) data integration from real agricultural multi datasets; (3) data science and business intelligent support; (4) high performance; (5) high storage; (6) security; (7) governance and monitoring; (8) consistency, availability and partition tolerant; (9) distributed and cloud deployment. We also evaluate the performance of our data warehouse.

Keywords: Data warehouse · Big Data · Precision agriculture

1 Introduction

In 2017 and 2018, annual world cereal productions were 2,608 million tons [30] and 2,595 million tons [7], respectively. However, there were also around 124 million people in 51 countries faced food crisis and food insecurity [8]. According to United Nations [29], we need an increase 60% of cereal production to meet 9.8 billion people needs by 2050. To satisfy the massively increase demand for food, crop yields must be significantly increased by using new farming approaches, such as precision agriculture. As reported in [6], precision agriculture is vitally important for the future and can make a significant contribution to food security and safety.

The precision agriculture's current mission is to use the decision-support system based on Big Data approaches to provide precise information for more control of farming efficiency and waste, such as awareness, understanding, advice, early warning, forecasting and financial services. An efficient agricultural data warehouse (DW) is required to extract useful knowledge and support decision-making. However, currently there are very few reports in the literature that

K. Chen et al. (Eds.): BigData 2019, LNCS 11514, pp. 1–17, 2019.
https://doi.org/10.1007/978-3-030-23551-2_1

focus on the design of efficient DWs with the view to enable Agricultural Big Data analysis and mining. The design of large scale agricultural DWs is very challenging. Moreover, the precision agriculture system can be used by different kinds of users at the same time, for instance by both farmers and agronomists. Every type of user needs to analyse different information sets thus requiring specific analytics. The agricultural data has all the features of Big Data:

1. Volume: The amount of agricultural data is rapidly increasing and is intensively produced by endogenous and exogenous sources. The endogenous data is collected from operation systems, experimental results, sensors, weather stations, satellites and farm equipment. The systems and devices in the agricultural ecosystem can connect through IoT. The exogenous data concerns the external sources, such as farmers, government agencies, retail agronomists and seed companies. They can help with information about local pest and disease outbreak tracking, crop monitoring, market accessing, food security, products, prices and knowledge.
2. Variety: Agricultural data has many different forms and formats, such as structured and unstructured data, video, imagery, chart, metrics, geo-spatial, multi-media, model, equation and text.
3. Velocity: The produced and collected data increases at high rate, as sensing and mobile devices are becoming more efficient and cheaper. The datasets must be cleaned, aggregated and harmonised in real-time.
4. Veracity: The tendency of agronomic data is uncertain, inconsistent, ambiguous and error prone because the data is gathered from heterogeneous sources, sensors and manual processes.

In this research, firstly, we analyze popular DWs to handle agricultural Big Data. Secondly, an agricultural DW is designed and implemented by combining Hive, MongoDB, Cassandra, and constellation schema on real agricultural datasets. Our DW has enough main features of a DW for agricultural Big Data. These are: (1) high storage, high performance and cloud computing adapt for the volume and velocity features; (2) flexible schema and integrated storage structure to adapt the variety feature; (3) data ingestion, monitoring and security adapt for the veracity feature. Thirdly, the effective business intelligent support is illustrated by executing complex HQL/SQL queries to answer difficult data analysis requests. Besides, an experimental evaluation is conducted to present good performance of our DW storage. The rest of this paper is organised as follows: in the next Section, we reviewed the related work. In Sects. 3, 4, and 5, we presented solutions for the above goals, respectively. Finally, Sect. 6 gives some concluding remarks.

2 Related Work

Data mining can be used to design an analysis process for exploiting big agricultural datasets. Recently, many papers have been published that exploit machine learning algorithms on sensor data and build models to improve agricultural economics, such as [23–25]. In these, the paper [23] predicted crop yield by using

self-organizing-maps supervised learning models; namely supervised Kohonen networks, counter-propagation artificial networks and XY-fusion. The paper [24] predicted drought conditions by using three rule-based machine learning; namely random forest, boosted regression trees, and Cubist. Finally, the paper [25] predicted pest population dynamics by using time series clustering and structural change detection which detected groups of different pest species. However, the proposed solutions are not satisfied the problems of agricultural Big Data, such as data integration, data schema, storage capacity, security and performance.

From a Big Data point of view, the papers [14] and [26] have proposed "smart agricultural frameworks". In [14], the platform used Hive to store and analyse sensor data about land, water and biodiversity which can help increase food production with lower environmental impact. In [26], the authors moved toward a notion of climate analytics-as-a-service by building a high-performance analytics and scalable data management platform which is based on modern infrastructures, such as Amazon web services, Hadoop and Cloudera. However, the two papers did not discuss how to build and implement a DW for a precision agriculture.

Our approach is inspired by papers [20,27,28] and [19] which presented ways of building a DW for agricultural data. In [28], the authors extended entity-relationship model for modelling operational and analytical data which is called the multi-dimensional entity-relationship model. They introduced new representation elements and showed the extension of an analytical schema. In [27], a relational database and an RDF triple store, were proposed to model the overall datasets. In that, the data are loaded into the DW in RDF format, and cached in the RDF triple store before being transformed into relational format. The actual data used for analysis was contained in the relational database. However, as the schemas in [28] and [27] were based on entity-relationship models, they cannot deal with high-performance, which is the key feature of a data warehouse.

In [20], a star schema model was used. All data marts created by the star schemas are connected via some common dimension tables. However, a star schema is not enough to present complex agricultural information and it is difficult to create new data marts for data analytics. The number of dimensions of DW proposed by [20] is very small; only 3-dimensions – namely, Species, Location, and Time. Moreover, the DW concerns livestock farming. Overcoming disadvantages of the star schema, the paper [19] proposed a constellation schema for an agricultural DW architecture in order to facilitate quality criteria of a DW. However, it did not describe how to implement the proposed DW. Finally, all papers [19,20,27,28] did not used Hive, MongoDB or Cassandra in their proposed DWs.

3 Analyzing Cassandra, MongoDB and Hive in Agricultural Big Data

In general, a DW is a federated repository for all the data that an enterprise can collect through multiple heterogeneous data sources belonging to various

enterprise's business systems or external inputs [9,13]. A quality DW should adapt many important criteria [1,15], such as: (1) Making information easily accessible; (2) Presenting and providing right information at the right time; (3) Integrating data and adapting to change; (4) Achieving tangible and intangible benefits; (5) Being a secure bastion that protects the information assets; and (6) Being accepted by DW users. So, to build an efficient agricultural DW, we need to take into account these criteria.

Currently, there are many popular databases that support efficient DWs, such as such as Redshift, Mesa, Cassandra, MongoDB and Hive. Hence, we are analyzing the most popular and see which is the best suited for our data problem. In these databases, Redshift is a fully managed, petabyte-scale DW service in the cloud which is part of the larger cloud-computing platform Amazon Web Services [2]. Mesa is highly scalable, petabyte data warehousing system which is designed to satisfy a complex and challenging set of users and systems requirements related to Google's Internet advertising business [10]. However, Redshift and Mesa are not open source. While, Cassandra, MongoDB and Hive are open source databases, we want to use them to implement agriculture DW. Henceforth, the Cassandra and MongoDB terms are used to refer to DWs of Cassandra and MongoDB databases.

There are many papers studying Cassandra, MongoDB and Hive in the view of general DWs. In the following two subsections, we present advantages, disadvantages, similarities and differences between Cassandra, MongoDB and Hive in the context of agricultural DW. Specially, we analyze to find how to combine these DWs together to build a DW for agricultural Big Data, not necessarily best DW.

3.1 Advantages and Disadvantages

Cassandra, MongoDB and Hive are used widely for enterprise DWs. Cassandra[1] is a distributed, wide-column oriented DW from Apache that is highly scalable and designed to handle very large amounts of structured data. It provides high availability with no single point of failure, tuneable and consistent. Cassandra offers robust support for transactions and flexible data storage based on ideas of DynamoDB and BigTable [11,18]. While, MongoDB[2] is a powerful, cross-platform, document oriented DW that provides, high performance, high availability, and scalability [4,12]. It works on concept of collection and document, JSON-like documents, with dynamic schemas. So, documents and data structure can be changed over time. Secondly, MongoDB combines the ability to scale out with features, such as ad-hoc query, full-text search and secondary index. This provides powerful ways to access and analyze datasets.

Hive[3] is an SQL data warehouse infrastructure on top of Hadoop[4] for writing and running distributed applications to summarize Big Data [5,16]. Hive can

[1] http://cassandra.apache.org.
[2] http://mongodb.com.
[3] http://hive.apache.org.
[4] http://hadoop.apache.org.

be used as an online analytical processing (OLAP) system and provides tools to enable data extract - transform - load (ETL). Hive's metadata structure provides a high-level, table-like structure on top of HDFS (Hadoop Distributed File System). That will significantly reduce the time to perform semantic checks during the query execution. Moreover, by using Hive Query Language (HQL), similar to SQL, users can make simple queries and analyse the data easily.

Although, the three DWs have many advantages and have been used widely, they have major limitations. These limitations impact heavily on their use as agricultural DW.

1. In Cassandra: (1) Query Language (CQL) does not support joint and subquery, and has limited support for aggregations that are difficult to analyze data; (2) Ordering is done per-partition and specified at table creation time. The sorting of thousands or millions of rows can be fast in development but sorting billion ones is a bad idea; (3) A single column value is recommended not be larger than 1 MB that is difficult to contain videos or high quality images, such as LiDAR images, 3-D images and satellite images.
2. In MongoDB: (1) The maximum BSON document size is 16 MB that is difficult to contain large data such as video, audio and high quality image; (2) JSON's expressive capabilities are limited because the only types are null, boolean, numeric, string, array, and object; (3) We cannot automatically rollback more than 300 MB of data. If we have more than that, manual intervention is needed.
3. Hive is not designed for: (1) Online transaction processing; (2) Real-time queries; (3) Large data on network; (4) Trivial operations; (5) Row-level update; and (6) Iterative execution.

3.2 Feature Comparison

Table 1 lists technical features used to compare Hive, MongoDB and Cassandra. For the ten overview features given in section A of Table 1, the three DWs differ in data schema, query language and access methods. However, they all support map reduce. Moreover, the ETL feature is supported by Hive, limited to Cassandra and unsupported by MongoDB. The full-text search feature is only supported by MongoDB. The secondary index and ad-hoc query features are supported by Hive and MongoDB but not or restricted by Cassandra. The 9^{th} feature being the Consistency – Availability – Partition tolerant classification (CAP) theorem says how the database system behaves when facing network instability. It implies that in the presence of a network partition, one has to choose between consistency and availability. Hive and Cassandra choose availability. While, MongoDB chooses consistency. Finally, the structure of Hive and MongoDB are master - slave while Cassandra has peer - to - peer structure.

The section B of Table 1 describes five industrial features, such as governance, monitoring, data lifecycle management, workload management, and replication-recovery. All of Hive, MongoDB and Cassandra support these features. Hive supports governance and data lifecycle management features via Hadoop. Cassandra is based on Java Management Extensions (JME) for governance.

Table 1. Technical features

No.	Features	Hive	MongoDB	Cassandra
A. Overview Features				
1	Data scheme	Yes	No-Schema	Flexible Schema
2	Query language	HQL	JS-like syntax	CQL
3	Accessing method	JDBC, ODBC, Thrift	JSON	Thrift
4	Map reduce	Yes	Yes	Yes
5	ETL	Yes	No	Limited
6	Full-text search	No	Yes	No
7	Ad-hoc query	Yes	Yes	No
8	Secondary index	Yes	Yes	Restricted
9	CAP	AP	CP	AP
10	Structure	Master – Slave	Master – Slave	Peer – to – Peer
B. Industrial Features				
1	Governance	Yes (via Hadoop)	Yes	Yes (via JME)
2	Monitoring	Yes	Yes	Yes
3	Data lifecycle management	Yes (via Hadoop)	Yes	Yes
4	Workload management	Yes	Yes	Yes
5	Replication-Recovery	Yes	Yes	Yes

The data management and DW features are described in section A and section B of Table 2, respectively. The data management features are security, high storage capacity, and data ingestion and pre-processing. The DWs have support for these features, except Cassandra does not support for data ingestion and pre-processing. Hive has the best for high storage capacity. The DW features are business intelligent, data science and high performance. Hive supports well business intelligent and data science but it is not suitable for real-time performance. MongoDB is very fast but it is limited in supporting for business intelligent and data science. Cassandra also is very fast and supports business intelligent but has limited capabilities for data science.

Table 2. Data Management and Data Warehouse Features

No.	Features	Hive	MongoDB	Cassandra
A. Data Management Features				
1	Security	Yes	Yes	Yes
2	High storage capacity	Yes (best)	Yes	Yes
3	Data ingestion and pre-processing	Yes	Yes	No
B. Data Warehouse Features				
1	Business intelligent	Very good	Limited	Good
2	Data science	Very good	Limited	Limited
3	High performance	Non-real time	Real time	Real time

4 Agricultural Data Warehouse

The general architecture of a typical DW includes four separate and distinct modules being Raw Data, ETL, Integrated Information and Data Mining. In the scope of this paper, we focus on the Integrated Information module which is a logically a centralised repository. It includes DW storage, data marts, data cubes and OLAP engine.

The DW storage is organised, stored and accessed using a suitable schema defined in the metadata. It can be either directly accessed or used to creating data marts which is usually oriented to a particular business function or enterprise department. A data cube is a data structure that allows fast analysis of data according to the multiple dimensions that define a business problem. The data cubes are created by the OLAP engine.

4.1 OLAP

OLAP is a category of software technology that provides the insight and understanding of data in multiple dimensions through fast, consistent, interactive access to enable analysts or managers to make better decisions. By using roll-up, drill-down, slice-dice and pivot operations, OLAP performs multidimensional analysis in a wide variety of possible views of information that provide complex calculations, trend analysis and sophisticated data modelling with a short execution time. So, OLAP is a key way to exploit information in a DW to allow end-users to analyze and explore data in multidimensional views.

The OLAP systems are categorised into three types: namely relational OLAP (ROLAP), multidimensional OLAP (MOLAP) and hybrid OLAP (HOLAP). In our agricultural Big Data context, HOLAP is more suitable than ROLAP and MOLAP because:

1. ROLAP has quite slow performance. Each ROLAP report is an SQL query in the relational database that requires a significant execution time. In addition, ROLAP does not meet all the users' needs, especially complex queries.
2. MOLAP requires that all calculations should be performed during the data cube construction. So, it handles only a limited amount of data and does not scale well. In addition, MOLAP is not capable of handling detailed data.
3. HOLAP inherits relational technique of ROLAP to store large data volumes and detailed information. Additionally, HOLAP also inherits multidimensional techniques of MOLAP to perform complex calculations and has good performance.

4.2 The Proposed Architecture

Based on the analyis in Sect. 3, Hive is chosen for building our DW storage and it is combining with MongoDB to implement our Integrated Information module. So, Hive contains database created from the our DW schema in the initialization period. This is for the following reasons:

1. Hive is based on Hadoop which is the most powerful tool of Big Data. Besides, HQL is similar to SQL which is familiar to the majority of users. Especially, Hive supports well high storage capacity, business intelligent and data science more than MongoDB and Cassandra. These features of Hive are useful to make an agricultural DW and apply data mining technologies.
2. Hive does not have real-time performance so it needs to be combined with MongoDB or Cassandra to improve performance of our Integrated Information module.
3. MongoDB is more suitable than Cassandra to complement Hive because: (1) MongoDB supports joint operation, full text search, ad-hoc query and second index which are helpful to interact with users. While Cassandra does not support these features; (2) MongoDB has the same master – slave structure with Hive that is easy to combine. While the structure of Cassandra is peer - to - peer; (3) Hive and MongoDB are more reliable and consistent. So the combination between Hive and MongoDB supports fully the CAP theorem while Hive and Cassandra are the same AP systems.

Our DW architecture for agricultural Big Data is illustrated in Fig. 1 which contains three modules, namely Integrated Information, Products and Raw Data. The Integrated Information module includes two components being MongoDB component and Hive component. Firstly, the MongoDB component will receive real-time data, such as user data, logs, sensor data or queries from Products module, such as web application, web portal or mobile app. Besides, some results which need to be obtained in real-time will be transferred from the MongoDB to Products. Second, the Hive component will store the online data from and send the processed data to the MongoDB module. Some kinds of queries having

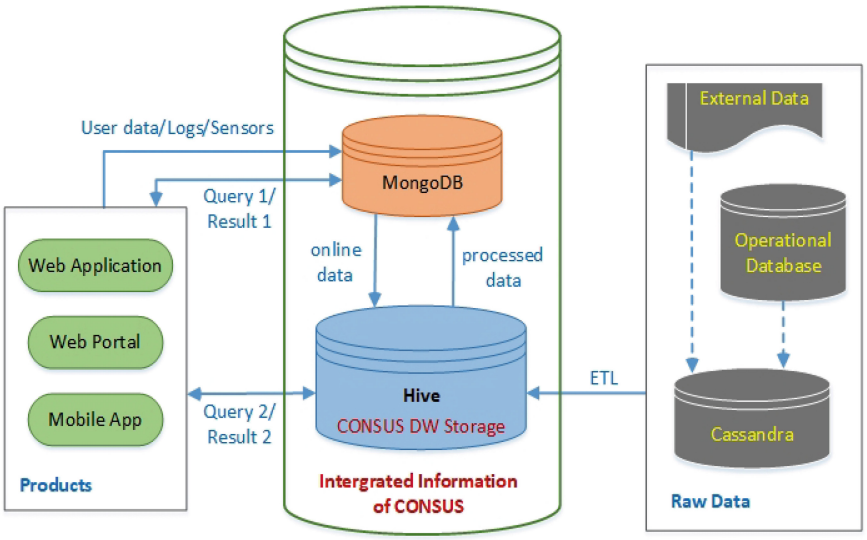

Fig. 1. Our agricultural data warehouse architecture

complex calculations will be sent directly to Hive. After that, Hive will send the results directly to the Products module.

In Raw Data module, almost data in Operational Databases or External Data components is loaded into Cassandra component. It means that we use Cassandra to represent raw data storage. In the idle times of the system, the update raw data in Cassandra will be imported into Hive through the ELT tool. This improves the performance of ETL and helps us deploy our system on cloud or distributed systems better.

4.3 Our Schema

The DW uses schema to logically describe the entire datasets. A schema is a collection of objects, including tables, views, indexes, and synonyms which consist of some fact and dimension tables [21]. The DW schema can be designed through the model of source data and the requirements of users. There are three kind of

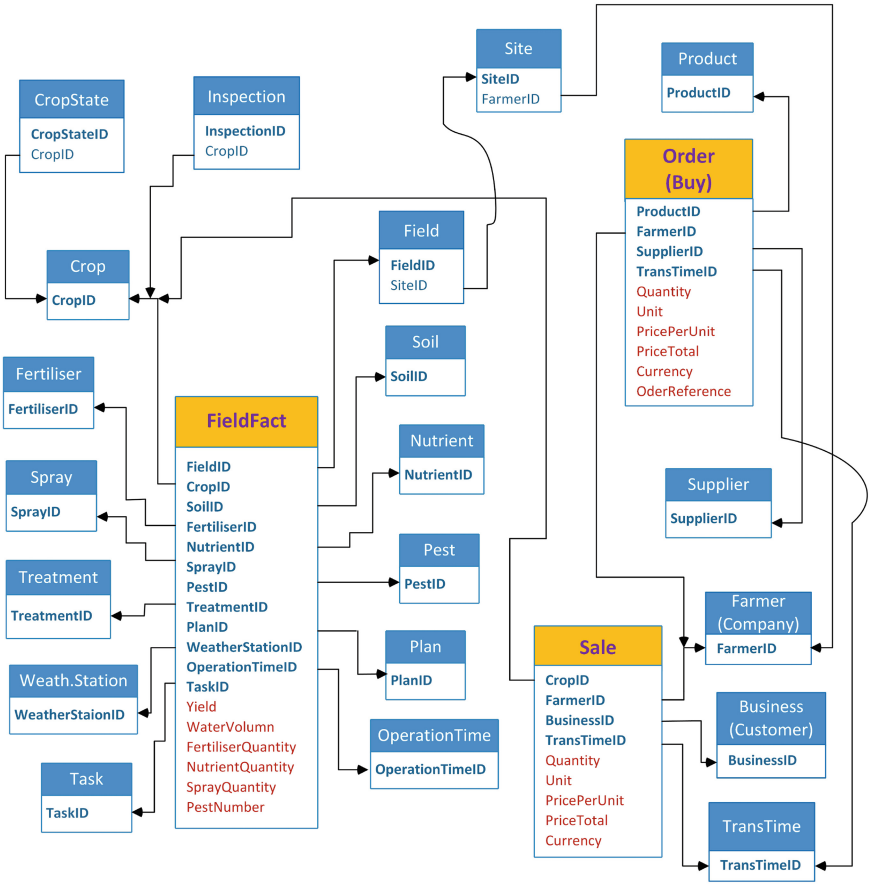

Fig. 2. A part of our data warehouse schema for Precision Agriculture

schemas, namely star, snowflake and constellation. With features of agricultural data, the agricultural DW schema needs to have more than one fact table and be flexible. So, the constellation schema, also known galaxy schema, is selected to design our DW schema.

We developed a constellation schema for our agricultural DW and it is partially described in Fig. 2. It includes 3 fact tables and 19 dimension tables. The FieldFact fact table contains data about agricultural operations on fields. The Order and Sale fact tables contain data about farmers' trading operations. The FieldFact, Order and Sale facts have 12, 4 and 4 dimensions, and have 6, 6 and 5 measures, respectively. While, dimension tables contain details about each instance of an object involved in a crop yield.

The main attributes of these dimension tables are described in the Table 3. The key dimension tables are connected to their fact table. However, there are some dimension tables connected to more than one fact table, such as Crop and Farmer. Besides, the CropState, Inspection and Site dimension tables are not connected to any fact table. The CropState and Inspection tables are used to support the Crop table. While, the Site table supports the Field table.

Table 3. Descriptions of some dimension tables

No.	Dim. tables	Particular attributes
1	Business	BusinessID, Name, Address, Phone, Mobile, Email
2	Crop	CropID, CropName, VarietyID, VarietyName, EstYield, SeasontSart, SeasonEnd, BbchScale, ScientificName, HarvestEquipment, EquipmentWeight
3	CropState	CropStateID, CropID, StageScale, Height, MajorStage, MinStage, MaxStage, Diameter, MinHeight, MaxHeight, CropCoveragePercent
4	Farmer	FarmerID, Name, Address, Phone, Mobile, Email
5	Fertiliser	FertiliserID, Name, Unit, Status, Description, GroupName
6	Field	FieldID, Name, SiteID, Reference, Block, Area, WorkingArea, FieldGPS, Notes
7	Inspection	InspectionID, CropID, Description, ProblemType, Severity, ProblemNotes, AreaValue, AreaUnit, Order, Date, Notes, GrowthStage
8	Nutrient	NutrientID, NutrientName, Date, Quantity
9	OperationTime	OperationTimeID, StartDate, EndDate, Season
10	Pest	PestID, CommonName, ScientificName, PestType, Description, Density, MinStage, MaxStage, Coverage, CoverageUnit
11	Plan	PlanID, PName, RegisNo, ProductName, ProductRate, Date, WaterVolume
12	Product	ProductID, ProductName, GroupName
13	Site	SiteID, FarmerID, SName, Reference, Country, Address, GPS, CreatedBy
14	Spray	SprayID, SProductName, ProductRate, Area, Date, WaterVol, ConfDuration, ConfWindSPeed, ConfDirection, ConfTemp, ConfHumidity, ActivityType
15	Soil	SoilID, PH, Phosphorus, Potassium, Magnesium, Calcium, CEC, Silt, Clay, Sand, TextureLabel, TestDate
16	Supplier	SupplierID, Name, ContactName, Address, Phone, Mobile, Email
17	Task	TaskID, Desc, Status, TaskDate, TaskInterval, CompDate, AppCode
18	Treatment	TreatmentID, TreatmentName, FormType, LotCode, Rate, ApplCode, LevlNo, Type, Description, ApplDesc, TreatmentComment
19	WeatherStation	WeatherStationID, Name, MeasureDate, AirTemp, SoilTemp, WindSpeed

5 Experiments

Through the proposed architecture in Sect. 4.2, our DW inherited many advantages from Hive, MongoDB and Cassandra presented in Sect. 3, such as high performance, high storage, large scale analytic and security. In the scope of this paper, we evaluated our DW schema and data analysis capacity on real agricultural datasets through complex queries. In addition, the time performance of our agricultural DW storage was also evaluated and compared to MySQL on many particular built queries belonging to different query groups.

Fig. 3. Data in UK and Ireland [22]

5.1 Data Analyzing Demo

The input data for the DW was primarily obtained from an agronomy company which supplies data from its operational systems, research results and field trials. Specially, we are supplied real agricultural data in iFarms, B2B sites, technology centres and demonstration farms. Their specific positions in several European countries are presented in Figs. 3 and 4 [22]. There is a total of 29 datasets. On average, each dataset contains 18 tables and is about 1.4 GB in size. The source datasets are loaded on our CONSUS DW Storage based on the schema described in Sect. 4.3 through an ETL tool. From the DW storage, we can extract and analyze useful information through tasks using complex HQL queries or data mining algorithms. These tasks could not be executed if the separate 29 datasets have not been integrated into our DW storage.

Fig. 4. Data in Continental Europe [22]

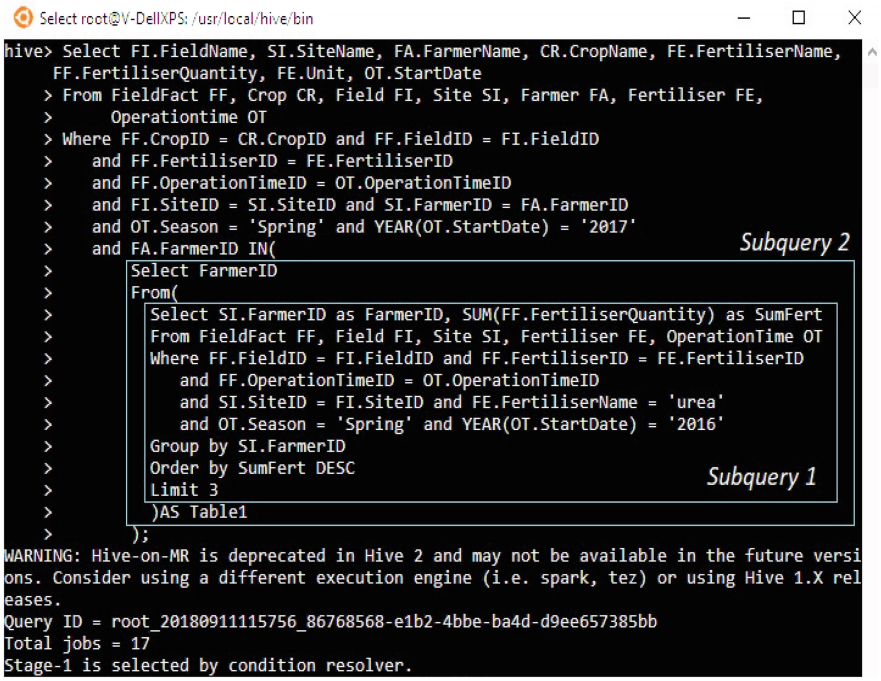

Fig. 5. A screenshort of executing the query example in our Hive

An example for a complex request: *"List crops, fertilisers, corresponding fertiliser quantities in spring, 2017 in every field and site of 3 farmers (crop companies) who used the large amount of Urea in spring, 2016"*. In our schema, this query can be executed by a HQL/SQL query as shown in Fig. 5. To execute this request, the query needs to exploit data in the FieldFact fact table and the six dimension tables, namely Crop, Field, Site, Farmer, Fertiliser and OperationTime. The query consists of two subqueries which return *3 farmers (crop companies) that used the largest amount of Urea in spring, 2016*.

5.2 Performance Analysis

The performance analysis was implemented using MySQL 5.7.22, JDK 1.8.0_171, Hadoop 2.6.5 and Hive 2.3.3 which run on Bash on Ubuntu 16.04.2 on Windows 10. All experiments were run on a laptop with an Intel Core i7 CPU (2.40 GHz) and 16 GB memory. We only evaluate reading performance of our DW storage because a DW is used for reporting and data analysis. The database of our storage is duplicated into MySQL to compare performance. By combining popular HQL/SQL

commands, namely Where, Group by, Having, Left (right) Join, Union and Order by, we create 10 groups for testing. Every group has 5 queries and uses one, two or more commands (see Table 4). Besides, every query also uses operations, such as And, Or, ≥, Like, Max, Sum and Count, to combine with the commands.

All queries were executed three times and we took the average value of the these executions. The different times in runtime between MySQL and our storage of query q_i is calculated as $Times_{q_i} = RT_{q_i}^{mysql}/RT_{q_i}^{ours}$. Where, $RT_{q_i}^{mysql}$ and $RT_{q_i}^{ours}$ are respectively average runtimes of query q_i on MySQL and our storage. Besides, with each group G_i, the different times in runtime between MySQL and our storage $Times_{G_i} = RT_{G_i}^{mysql}/RT_{G_i}^{ours}$. Where, $RT_{G_i} = Average(RT_{q_i})$ is average runtime of group G_i on MySQL or our storage.

Table 4. Command combinations of queries

Group	Queries	Where	Group by	Having	Left (right) Joint	Union	Order by
1	1–5	x					
2	6–10	x	x				
3	11–15	x			x		
4	16–20	x				x	
5	21–25	x					x
6	26–30	x			x		x
7	31–35	x	x	x			
8	36–40	x	x	x			x
9	41–45	x	x	x	x		x
10	45–50	x	x	x		x	x

Figure 6 describes different times between MySQL and our storage in runtime of every query belongs to 10 groups. Unsurprisingly, although running on one computer, but with large data volume, our storage is faster than MySQL at 46/50 queries and all 10 query groups. MySQL is faster than our storage at 3 queries 12^{th}, 13^{th} and 18^{th} belonging to groups 3^{rd} and 4^{th}. Two databases are same at the query 25^{th} belonging to group 5^{th}. Within each query group, to have a fair performance comparison, the queries combine randomly fact tables and dimensional tables. This makes the complex of queries having far high difference. Combining with different size and structure of the tables, it make the runtime of queries being huge differences although belonging a group, as presented in Fig. 6.

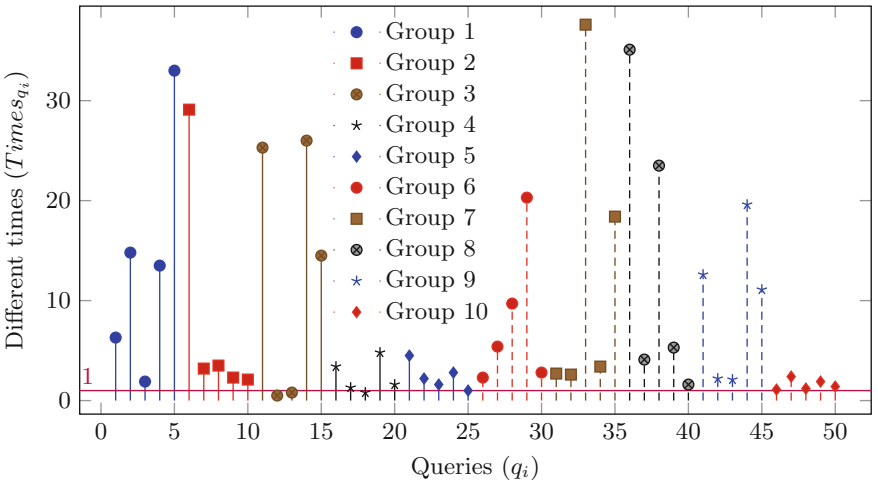

Fig. 6. Different times between MySQL and our storage in runtime of every Query

Beside comparing runtime in every query, we aslo compare runtime of every group presented in Fig. 7. Comparing to MySQL, our storage is more than at most (6.24 times) at group 1^{st} which uses only *Where* command, and at least (1.22 times) at group 3^{rd} which uses *Where* and *Joint* commands.

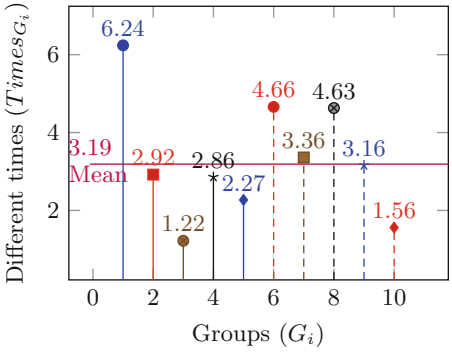

Fig. 7. Different times between MySQL and our storage in runtime of every group

Figure 8 presents the average runtime of the 10 query groups on MySQL and our storage. Mean, the run time of a reading query on MySQL and our storage is 687.8 s and 216.1 s, respectively. It means that our storage is faster 3.19 times. In the future, by deploying our storage solution on cloud or distributed systems, we believe that the performance will be even much better than MySQL.

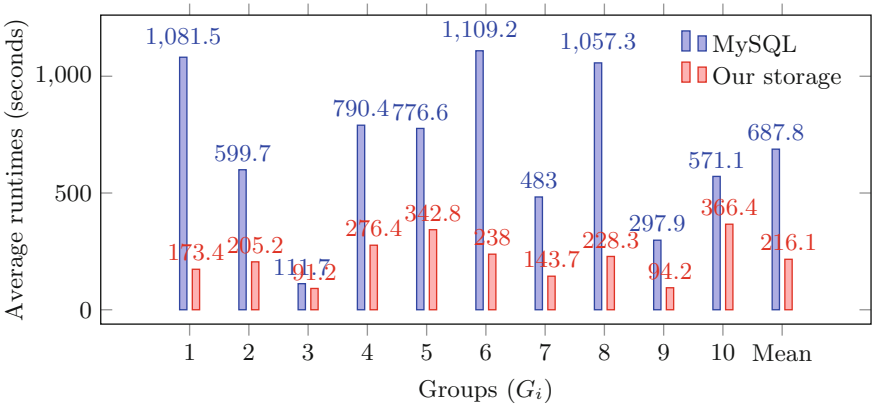

Fig. 8. Average Runtimes of MySQL and our storage in every Groups

6 Conclusion and Future Work

In this paper, we compared and analyzed some existing popular open source DWs in the context of agricultural Big Data. We designed and implemented the agricultural DW by combining Hive, MongoDB and Cassandra DWs to exploit their advantages and overcome their limitations. Our DW includes necessary modules to deal with large scale and efficient analytics for agricultural Big Data. Additionally, the presented schema herein was optimised for the real agricultural datasets that were made available to us. The schema been designed as a constellation so it is flexible to adapt to other agricultural datasets and quality criteria of agricultural Big Data. Moreover, using the short demo, we outlined a complex HQL query that enabled knowledge extraction from our DW to optimize of agricultural operations. Finally, through particular reading queries using popular HQL/SQL commands, our DW storage outperforms MySQL by far.

In the future work, we shall pursue the deployment of our agricultural DW on a cloud system and implement more functionalities to exploit this DW. The future developments will include: (1) Sophisticated data mining techniques [3] to determine crop data characteristics and combine with expected outputs to extract useful knowledge; (2) Predictive models based on machine learning algorithms; (3) An intelligent interface for data access; (4) Combination with the high-performance knowledge map framework [17].

Acknowledgment. This research is part of the CONSUS research programme which is funded under the SFI Strategic Partnerships Programme (16/SPP/3296) and is co-funded by Origin Enterprises Plc.

References

1. Adelman, S., Moss, L.: Data Warehouse Project Management, 1st edn. Addison-Wesley Professional, Boston (2000)

2. Amazon team: Amazon Redshift database developer guide. Samurai ML (2018)
3. Cai, F., et al.: Clustering approaches for financial data analysis: a survey. In: The 8th International Conference on Data Mining (DMIN 2012), pp. 105–111 (2012)
4. Chodorow, K.: MongoDB: The Definitive Guide. Powerful and Scalable Data Storage, 2nd edn. O'Reilly Media, New York (2013)
5. Du, D.: Apache Hive Essentials, 2nd edn. Packt Publishing (2018)
6. Eurobarometer team: Europeans, agriculture and the common agricultural policy. Special Eurobarometer 440, The European Commission (2016)
7. FAO-CSDB team: Global cereal production and inventories to decline but overall supplies remain adequate. Cereal Supply and Demand Brief, FAO, 06 December 2018
8. FAO-FSIN team: Global report on food crises 2018. Food Security Information Network, FAO (2018)
9. Golfarelli, M., Rizzi, S.: Data Warehouse Design: Modern Principles and Methodologies. McGraw-Hill Education, New York (2009)
10. Gupta, A., et al.: Mesa: a geo-replicated online data warehouse for Google's advertising system. Commun. ACM **59**(7), 117–125 (2016)
11. Hewitt, E., Carpenter, J.: Cassandra: The Definitive Guide. Distributed Data at Web Scale, 2nd edn. O'Reilly Media, New York (2016)
12. Hows, D., et al.: The Definitive Guide to MongoDB. A Complete Guide to Dealing with Big Data Using MongoDB, 3rd edn. Apress, Berkely (2015)
13. Inmon, W.H.: Building the Data Warehouse. Wiley, New York (2005)
14. Kamilaris, A., et al.: Estimating the environmental impact of agriculture by means of geospatial and big data analysis. Science to Society, pp. 39–48 (2018)
15. Kimball, R., Ross, M.: The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling, 3rd edn. Wiley, New York (2013)
16. Lam, C.P., et al.: Hadoop in Action, 2nd edn. Manning, Greenwich (2016)
17. Le-Khac, N.-A., et al.: Distributed knowledge map for mining data on grid platforms. Int. J. Comput. Sci. Network Secur. **7**(10), 98–107 (2007)
18. Neeraj, N.: Mastering Apache Cassandra, 2nd edn. Packt Publishing, Birmingham (2015)
19. Ngo, V.M., et al.: An efficient data warehouse for crop yield prediction. In: The 14th International Conference Precision Agriculture (ICPA-2018), pp. 3:1–3:12 (2018)
20. Nilakanta, S., et al.: Dimensional issues in agricultural data warehouse designs. Comput. Electron. Agric. **60**(2), 263–278 (2008)
21. Oracle team: Database data warehousing guide, Oracle12c doc release 1 (2017)
22. Origin team: Annual report and accounts, Origin Enterprises plc (2018)
23. Pantazi, X.E.: Wheat yield prediction using machine learning and advanced sensing techniques. Comput. Electron. Agric. **121**, 57–65 (2016)
24. Park, S., et al.: Drought assessment and monitoring through blending of multisensor indices using machine learning approaches for different climate regions. Agric. Forest Meteorol. **216**, 157–169 (2016)
25. Rupnik, R., et al.: AgroDSS: a decision support system for agriculture and farming. Computers and Electronics in Agriculture (2018)
26. Schnase, J., et al.: MERRA analytic services: meeting the big data challenges of climate science through cloud-enabled climate analytics-as-a-service. Comput. Environ. Urban Syst. **61**, 198–211 (2017)
27. Schuetz, C.G., et al.: Building an active semantic data warehouse for precision dairy farming. Organ. Comput. Electron. Commerce **28**(2), 122–141 (2018)

28. Schulze, C., et al.: Data modelling for precision dairy farming within the competitive field of operational and analytical tasks. Comput. Electron. Agric. **59**(1–2), 39–55 (2007)
29. UN team: World population projected to reach 9.8 billion in 2050, and 11.2 billion in 2100. Department of Economic and Social Affairs, United Nations (2017)
30. USDA report: World agricultural supply and demand estimates 08/2018. United States Department of Agriculture (2018)