2015 IEEE First International Conference on Big Data Computing Service and Applications

Towards a 'Big' Health Data Analytics Platform

Sangwhan Cha, Ashraf Abusharekh, Syed SR Abidi
NICHE Research Group, Faculty of Computer Science
Dalhousie University
Halifax, Canada
sangwhan.cha@gmail.com, asharf@dal.ca, ssrabidi@dal.ca

***Abstract*— Health is generating large volumes of data that can provide invaluable insights into clinical and operational aspects of healthcare delivery. There is a general lack of specialized and integrated health data analytics platforms that offer technical methods to support the entire health data analysis pipeline—i.e. health data selection, integration, analysis, visualization and sharing. This paper proposes the technical architecture of a health data analytics platform that offers a technical solution for analyzing 'big' health data originating from multiple sources with heterogeneous terminologies and schemas. A key aspect of the architecture is data standardization, where we have used SNOMED-CT as a terminology standard to standardize health data from multiple sources. We offer a single step health data integration solution where users can select the data sources and the data elements from multiple sources, and our platform performs the data standardization and data integration to prepare an integrated dataset. We present a case study involving large volumes of laboratory data that is integrated and analyzed using our platform.**

Keywords— Health informatics; Big health data; Health data analytics; Hadoop/MapReduce; SNOMED CT; data integration; data standardization

I. INTRODUCTION

Healthcare is generating a large amount of data given the initiatives of electronic medical records, personal health records, personalized medicine and clinical studies. There is an imminent need to analyze the health data in order to generate both clinical and operational intelligence that will impact health systems efficiency, patient safety and discovery of new interventions [1]. These massive quantities of health data, which can be regarded as 'big data', hold the promise of supporting a wide range of medical and healthcare functions, including clinical decision support, disease surveillance, and population health management when health data analysis is performed properly [13]. The analytical challenges that come with the availability of 'big' health data are many and can be summarized as (i) the integration of multi-modal data coming from different sources to generate a semantically consistent dataset; (ii) the interpretation of the data and the analysis from different stakeholder perspectives; (iii) the visualization of the analysis in timely, meaningful and customizable formats, and (iv) the sharing of the data and analysis for both research and clinical purposes.

Given the challenges pertaining to the analysis of big health data and the diversity of health data analysis needs, it is strategic to develop specialized data analytics platforms that incorporate a range of methods/components targeting the health data analysis pipeline—i.e. health data selection, integration, analysis, visualization and sharing. Furthermore, it is important that users of health data analytics platforms are able to perform their analytical tasks without the need to acquire a deeper understanding of the underlying technical methods. At this point there is a lack of dedicated and integrated 'big' health data analytics platforms, rather what is available are applications that perform a specific health data analytics task, and hence users are required to either use multiple applications or integrate these applications in order to perform health data analysis.

The literature outlines two main approaches for developing health data analytics platforms. The first approach advocates the use of open data analysis methods such as Mahoot [25] and open graphic APIs such as Google Charts [30]. This approach offers the flexibility of fully featured and more complex data analysis solutions using tools such as Mahout [25], Pig [33] and Hive [24]. The second approach is based on using a web application framework such as Shiny [31] that offers the functionality to rapidly develop web-accessible interactive health analytics platforms based on the statistical package R [32]. Both approaches target data analysis methods, yet they lack data integration interfaces that offer semantic interoperability between the data sources. Health data originates from multiple agencies/institution, each database with local clinical terms and database schemas. To generate big health data it is necessary to integrate these multiple health data sources which brings to relief the challenge of (a) data standardization to achieve a standard data interpretation; and (b) data integration to realize an integrated, multi-dimensional dataset. Health data standardization is achieved by leveraging a standard terminology systems (such as SNOMED, MESH) which should yield a complete, consistent and semantically interpretable dataset for downstream analysis. Integrated health data is regarded as big data, and for practical purposes current relational databases and data analysis methods based on standalone computing are not suitable for big health data analysis.

978-1-4799-8128-1/15 $31.00 © 2015 IEEE
DOI 10.1109/BigDataService.2015.13

Therefore, there is a need to investigate specialized integrated platforms for analyzing big health data..

In this paper we present the development of an integrated health data analytics platform that supports the entire lifecycle of health data analytics, spanning from data selection to integration to analysis to visualization. A key feature of the underlying design of our health data analytics platform is the integration of the two approaches for data analytics mentioned above, where we have two independent yet integrated data analytics modules—one module is based on open data analytics for complex analytical tasks and the other module leverages R libraries for interactive querying and analytical tasks. In particular, we focus on approaches for data standardization given that we need to integrate health data from multiple sources. Our primary focus in this paper is our solution for data standardization which is achieved in terms of semantic interoperability between the content and schemas of multiple databases. For establishing semantic interoperability at the data level (where data is stored in localized databases of health institutions), we use the Systemized Nomenclature of Medicine – Clinical Terms (SNOMED CT) which is a standard health care terminology [2,3]. For data integration, we employ a data warehousing architecture instead of a virtual data integration architecture. Our rationale is that a data warehousing architecture supports the data standardization and integration of the local data into a single standardized warehouse as compared to a virtual data integration architecture that needs to access data from localized medical institutions at query time. In terms of data warehouse we use the well known MapReduce paradigm with Hadoop Distributed File System (HDFS). We offer a single step data integration where users can select the data sources and the data elements (stored in separate tables) of interest from these sources. Our platform performs the data standardization and data integration to prepare an integrated dataset. Next, data selection function is available where users can select the integrated data for analysis using a range of analytical methods. For data visualization, we offer a range of web-based data visualizations. To illustrate the working of our health data analytics platform, we present a case study using laboratory data sourced as two separate local databases with different schema, where we demonstrate the entire process from data integration to data visualization.

The rest of the paper is structured as follows. Section 2 describes the model for building an efficient health data analytics platform. Section 3 describes the architectural view of the infrastructure and the various components of the system. The implementation of our prototype system is discussed in Section 4. Section 5 evaluates the prototype implementation and discusses the test results. Section 6 presents related work and section 7 gives concluding remarks.

II. MODEL FOR BUILDING AN EFFICIENT HEALTH DATA ANALYTICS PLATFORM

Data integration with semantic interoperability is done prior to data analysis. The output of data integration is called target dataset. Data stored in local databases of medical institutions are called source datasets. For the data integration component, there are three functionalities, (1) data selection, (2) data aggregation and (3) data uploading. Data selection is required when the analyst needs to select data among available source datasets for data aggregation. Data aggregation checks whether the selected data has already been uploaded. If not, aggregates the selected data into an intermediary target dataset. Finally the intermediary target dataset is uploaded to HDFS.

The data standardization component is responsible for semantic interoperability among medical terms used in different medical institutions. The data standardization component consists of (1) a data comparator and (2) a data converter. The data comparator compares each medical term in SNOMED CT corresponding to the local term with standard medical terms. The data converter converts the local term to the matching standard medical term.

After data is uploaded to HDFS, the analyst can use the data selection component to further slice and dice the target datasets to produce new ones. The data selection component consists of (1) data selection planner (2) data selection plan execution and (3) data generation. The data selection planner allows the analyst to select columns from different target datasets residing in HDFS. The data selection plan execution generates and executes a Hive query corresponding to data selection. The data generation generates a new dataset based on the results of data selection execution that becomes part of the target datasets.

The analyst can perform aggregation; data analyses and/or data export on target datasets uploaded or generated depending on his goals. Data analysis can be performed using tools and libraries such as Mahout, Pig, R, Hive, etc. Data export functionality is used to export the results of data analyses to the data visualization and reporting component.

For data analyses, and selection we are using R, Impala [35] and RImpala [37], for data visualization and reporting, we use Shiny, a web application framework for R.

III. SYSTEM ARCHITECTURE

The system architecture consists of three major subsystems. These sub-systems are shown in Figure 1. The DI (Data Integration) sub system is responsible for handling data integration with semantic interoperability. It handles data integration and data standardization. The DA (Data Analysis) handles health data analysis based on integrated data on HDFS resulting from the DI. It handles data selection, data analysis and data visualization using Hive. The EDA (Extended Data Analysis) allows the analyst to utilize the power of R and Impala in the process.

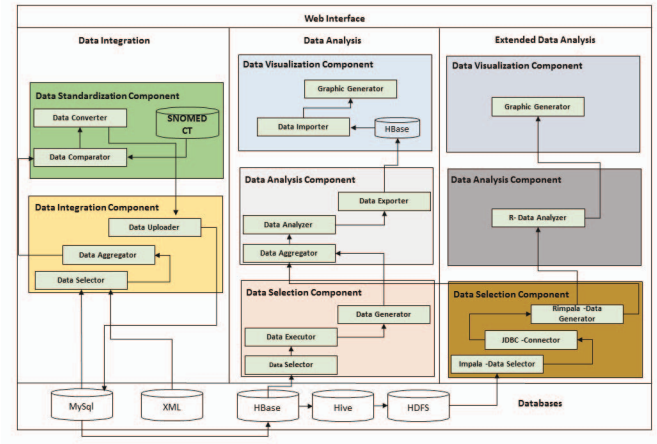
A. DI Subsystem

The DI subsystem of the system architecture consists of two components, (1) DSC (Data Standardization Component) and (2) DIC (Data Integration Component). The DSC is responsible for semantic interoperability among data stored locally at medical institutions during the data integration process using DC (Data Comparator) and DCON (Data Converter) functions. The DIC is responsible for data selection, data aggregation and data uploading. The DS (Data Selector) function of the DIC selects source datasets for data integration from an XML configuration data store. In addition to integration metadata, the XML configuration data store also contains connection and user credential information for all available external source datasets. After the DS selects source datasets from the XML configuration file, it notifies the DA (Data Aggregator) to aggregate data selected by the DS. Before the DA aggregates data, the DCON converts local medical terms present in source datasets into standard medical terms and forwards the converting result to the DA. The DC compares each searched SNOMED CT with terms in the standard term tables and provides the DCON with the matching term.

B. Data Analysis and Extended Data Analysis Subsystems

The DA subsystem of the system architecture consists of three components, which are DSPC (Data Selection Planner Component), DAC (Data Analysis Component) and DVC (Data Visualization Component). This subsystem is mainly responsible for data analysis and data visualization. The DSel (Data Selector) in the DSPC selects data stored in HBase linked with Hive store and invokes the DSE (Data Selection Executor) to execute the selection plan. The DSE executes Hive query with selected data resulting from the DSel and invokes the DG (Data Generator) to generate a new dataset from the selected data.

Fig. 1. System Architecture



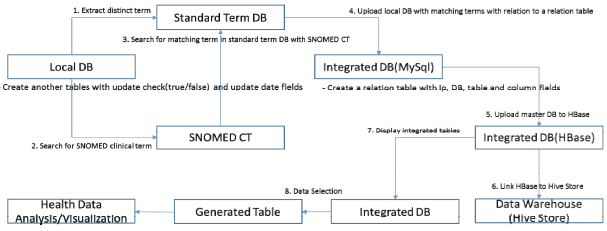
The DAGG (Data Aggregator) in the DAC can be applied to aggregate data from target datasets (integrated datasets and user-generated datasets). The DA performs data analysis on target datasets and invokes the DE (Data Exporter) to export the results to HBase. The GG (Graphic Generator) uses the Data Importer to extract data from HBase to generate visualizations and reports to the user.

To further enrich the analysis, visualization and reporting capabilities, the extended Data Analysis subsystem utilizes the Impala and RImpala for execution of distributed queries [40], and Shiny web framework for visualization and reports. Impala a massively parallel processing (MPP) SQL query engine that runs natively in Apache Hadoop [40], selects data stored on HBase and HDFS. RImpala is utilized to establish the connection between R and Impala through JDBC integration.

IV. IMPLEMENTATION OF HEALTH DATA ANALYTICS PLATFORM

To demonstrate the effectiveness of our proposed system architecture, we have implemented the prototype of a health data analytics platform mainly based on Hadoop/MapReduce. The prototype implementation makes the system design concrete by applying the following development environments: Spring MVC and Hibernate persistence as a framework, Maven as a building tool, Git as a version control, MySQL and HBase as databases, Hive Store as a data warehouse, Mahout and HQL as an analysis tools and Snowflake API as a SNOMED CT conversion. Fig. 2 illustrates the entire implementation phases with a brief explanation.

Fig. 2. Implementation Phases



For the first phase, we extract distinct terms from one of the local databases and store those terms in the standard term database. Before the data integration process, we create another table along with data compiled with a master database schema with additional attributes, which are an update check and an update date. For the second phase, the system searches for SNOMED CTs corresponding to each local clinical term. For the third phase, the system searches for matching terms in the standard term DB with searched SNOMED CTs. Before the fourth phase, we create a relation table with ip address, DB name and column name attributes for identifying the origination of data in the integrated database. Also, there are some rules for the fourth phase as follows:

-If one of SNOMED CT's converted terms is equal to one of the standard clinical terms in the standard terms database, it will be uploaded instead of a local clinical term.

-If more than two of SNOMED CT's converted terms are equal to some of the standard clinical terms in the standard term database, the term with the highest weight will be uploaded instead of a local clinical term.

-In the second case, if more than two of matched terms with the same highest weight exist, the first one will be uploaded instead of a local clinical term.
-If any of SNOMED CT converted terms is not equal to any of standard clinical terms in the standard terms database, the first highest term will be uploaded instead of a local clinical term and it will be added into the standard term database.

For the fourth phase, the system uploads the local DB into an integrated DB in MySQL with matching terms with relation to a relation table. We use an intermediary DB in MySQL for the integrated DB because of efficient schema mapping between MySQL and HBase. For the fifth phase, the system uploads the integrated DB into HBase. For the sixth phase, the system links HBase with Hive Store for further analysis.

For the seventh phase, the system displays integrated tables in the integrated DB for data selection. Finally, the system generates a result table based on selected data for data analysis and data visualization.

A. Data Standardization

Since SNOMED CT is a comprehensive reference terminology that allows healthcare providers to record clinical encounters accurately and unambiguously [22, 14-15], we decided to apply SNOMED CT for our proposed system. For data standardization, we developed a system that automatically converts a local medical term into SNOMED CT terminology. Data converting process occurs before data uploading into a temporary database. All medical terms used in a database schema are standardized using SNOMED CT terminology.

Developing a standard term database has to be prior to this process. There would be several SNOMED CT terminologies corresponding to each local clinical term. The system looks for a matching term in the standard term database corresponding to each SNOMED CT terminology. After that, the matching term will be uploaded into the integrated database through the temporary database.

Algorithm 1 explains the data integration with the procedure of a source instance conversion to a target instance in the standard term database. This algorithm starts with loading SST_{LCT} where SST is a source schema table and LCT is a local clinical term. A *listResult* is a list for storing the result from a *CompareStandardTerm().* If the attribute of the source schema table is composed of medical terms which need to be converted into SNOMED CT, this algorithm will be applied. A *DecideSSTAttribute()* is to decide whether the attribute of the source schema table needs to be converted into SNOMED CT or not. A *listSnomedCT* is for storing the result of a *GetSnomedCT()* which returns several SNOMED CT terminologies for $SST_{LCT.}$ For each returned SNOMED CT terminology, we need to decide whether there is the same term in the standard term database. If so, it will be added into the *listResult.* If there are more than zero elements in the *listResult,* the first element will be uploaded because the first element has the highest weight on similarity. Otherwise, the SST_{LCT} will be uploaded into the integrated database and inserted into the standard term database using a *UploadIntegaratedDB() and* a *InsertStandardTermDB()* respectively.

Algorithm 1 Data integration with data standardization
Input: SST_{LCT} /* SST -> Source Schema Table & LCT ->Local Clinical Term */
Output: TST_{SCT} /*TST ->Target Schema Table & SCT ->Standard Clinical Term */
begin

```
LoadSST_LCT();
Let listResult = null;
if (DecideSSTAttribute())
    Let listSnomedCT <- GetSnomedCT(SST_LCT)
      for SCT  listSnomedCT do
        if (CompareStandardTerm(SST_LCT))
          listResult.add(SST_LCT)
        end
        if listResult.size()>0
          UploadIntegaratedDB(listResult.get(0))
        else
          UploadIntegaratedDB(SST_LCT)
          InsertStandardTermDB(SST_LCT)
else
        UploadIntegaratedDB(SST_LCT)

Return TST_SCT
```

End

B. Data Integration

Since data integration deals with various local databases, it should resolve heterogeneities with respect to the schemas and their data [8]. Therefore, schema mapping with merging the schemas or computing the differences plays an important role in data integration [20]. Also, schema mappings are fundamental for a number of important information integration problems including data integration, data sharing, schema integration and data analysis [27].

We use a semi-automatic schema mapping approach, which means that the schema mapping information including merging information stored in XML configuration file as in File 1 and data integration is executed according to this file.

File 1. Database configuration for data integration

```
<?xml version="1.0" encoding="UTF-8"?>
<Scema>
 <global id='100' name="globalhealth" label="Global Health DB"
connect="/hibernate.cfg.xml">
 </global>
 <integrated id='99' name="integratedhealth" label="Integrate Health DB"
connect="/hibernate5.cfg.xml">

   <tb id='0' name="integratedlaborder" label="Integrated Lab Order"
type="order" clsName ="IntegratedLabOrder">
      <col name="LAB_ID" label="" paramName="labId" />
      <col name="REQUEST_TIMESTAMP" label=""
paramName="requestTimestamp" />
      <col name="TEST_TYPE" label="" paramName="testType" />
      <col name="CREATE_DATE" label="" paramName="createDate"/>
```

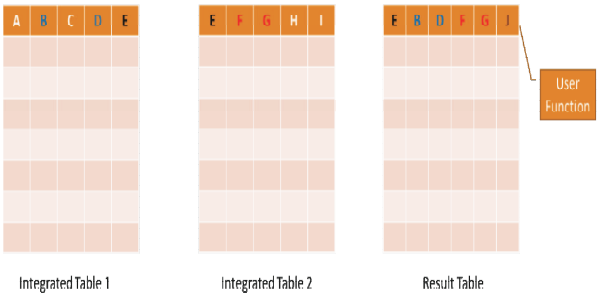
```
    </tb>
    <tb id='1' name="integratedresult" label="Integrated Lab Result"
type="order" clsName ="IntegratedResult">
        <col name="RESULT_ID" label="" paramName="resultId" />
        <col name="LAB_ID" label="" paramName="labId" />
        <col name="RESULT_TIMESTAMP" label=""
paramName="resultTimeStamp" />
        <col name="RESULT" label="" paramName="result" />
        <col name="CREATE_DATE" label="" paramName="createDate"/>
    </tb>
 </integrated>

 <db id='0' name="nslab" label="Nova Scotia Lab"
connect="/hibernate3.cfg.xml">
    <tb id='0' pid='0' name="laborder" label="Lab Order" type="order"
clsName ="NsLabOrderDetails" checked="1"
targetName="integratedlaborder" relationlocalId="1">
        <col name="ORDER_ID" label="" paramName="orderId"
targetName="labId" checked="0"/>
        <col name="ORDER_STATUS" label="" paramName="orderStauts"
targetName="" checked="0"/>
        <col name="CANCEL_REASON" label=""
paramName="cancelReason" targetName="" checked="0"/>
        <col name="ORDER_DATE" label="" paramName="orderDate"
targetName="requestTimestamp" checked="0"/>
        <col name="ORDER_TIME" label="" paramName="orderTime"
targetName="requestTimestamp" checked="0"/>
        <col name="ORDERABLE" label="" paramName="orderable"
targetName="testType" checked="1" />
    </tb>
    <tb id='1' pid='0' name="labresult" label="Lab Result" type="result"
clsName ="NsLabResultDetails" checked="0"
targetName="integratedresult" relationlocalId="1">
        <col name="RESULT_ID" label="" paramName="resultId"
targetName="resultId" checked="0"/>
        <col name="ORDER_ID" label="" paramName="orderId"
targetName="labId" checked="0"/>
        <col name="RESULT_STATUS" label=""
paramName="resultStatus" targetName="" checked="0"/>
        <col name="RESULT_DATE" label="" paramName="resultDate"
targetName="resultTimeStamp" checked="0"/>
        <col name="RESULT_TIME" label="" paramName="resultTime"
targetName="resultTimeStamp" checked="0"/>
        <col name="RESULT_TYPE" label="" paramName="resultType"
targetName="" checked="0"/>
        <col name="RESULT_VALUE_NUMERIC" label=""
paramName="resultValueN" targetName="result" checked="0"/>
    </tb>
 </db>
 <db id='1' name="onlab" label="Ontario Lab"
connect="/hibernate4.cfg.xml">
    <tb id='2' pid='1' name="onlaborder" label="" type="order" clsName
="ONLabOrderDetails" checked="1"  targetName="integratedlaborder"
relationlocalId="2">
        <col name="ID" label="" paramName="orderId" targetName="labId"
checked="0"/>
        <col name="ORDER_STATUS" label="" paramName="orderStauts"
targetName="" checked="0"/>
        <col name="REQUEST_TIMESTAMP" label=""
paramName="requestTimestamp" targetName="requestTimestamp"
checked="0"/>
        <col name="LAB_TYPE" label="" paramName="labType"
targetName="testType" checked="1"/>
    </tb>
    <tb id='3' pid='1' name="orderresult" label="" type="result" clsName
="ONLabResultDetails" checked="0"  targetName="integratedresult"
relationlocalId="2">
        <col name="ORDER_RESULT_ID" label="" paramName="resultId"
targetName="resultId" checked="0"/>
        <col name="ID" label="" paramName="orderId" targetName="labId"
checked="0"/>
        <col name="RESULT_STATUS" label=""
paramName="resultStatus" targetName="" checked="0"/>
        <col name="RESULT_TIMESTAMP" label=""
paramName="resultTimeStamp" targetName="resultTimeStamp"
checked="0"/>
        <col name="RESULT_TYPE" label="" paramName="resultType"
targetName="" checked="0"/>
        <col name="RESULT" label="" paramName="result"
targetName="result" checked="0"/>
    </tb>
 </db>
</Scema>
```

In this XML configuration file, we can obtain source databases, which is named as nslab and onlab and target database, which is named as integratedhealth, with table information.

C. Data Selection

In our proposed system, a user can generate a table consisting of some attributes from different tables for the data analysis and data visualization by selecting attributes on integrated tables. For example, attribute B and D from integrated table1 are selected and attribute F and G from integrated table 2 are selected as shown in figure 3, the result table will be generated with selected attributes. Also, other attributes can be added by inputting a user function such as diff., max., min. or avg..

Fig. 3. Example of Data Selection



Since the data selection process is executed through an Apache Hive, Hive Query Language (HQL) will be generated as shown as HQL 1. We assume that a user function is *diff(IntegratedTable1.B – IntegratedTable2.F)*.

HQL 1 Example of data selection

SELECT IntegratedTable1.B, IntegratedTable1.D, IntegratedTable2.F, IntegratedTable2.G, diff(IntegratedTable1.B – IntegratedTable2.F) FROM IntegratedTable1 JOIN IntegartedTable2 on (IntegratedTable1.E = IntegratedTable2.E)

This HQL is converted into a single MapReduce job as only E attribute on both tables are involved this join HQL [24].

V. EXPERIMENTS OF HEALTH DATA ANALYTICS PLATFORM IMPLEMENTATION

We conducted several experiments for the system prototype implementation. The experiments were chosen to provide an overview of both functional requirements such as data integration with data standardization, data selection, data analysis and data visualization as well as non-functional requirements such as application performance.

The objective of the first experiment was to see if our proposed system is able to perform data integration with semantic interoperability. For this experiment, we needed to observe several key points such as data merging and data validation after data integration with data standardization. The second experiment was to see if our proposed system is able to perform data selection using Hive query language based on data stored on HDFS and generate a table on HDFS (DA subsystem) so that the system can access the generated table for further analysis. The third experiment was to see if our proposed system was able to perform data selection, data analysis and data visualization using the Shiny web application framework for the EDA subsystem. Once the application was executed, the non-functional testing such as comparing the execution time for SQL join query statement and loading time for the graphic result can be performed.

The applications were executed under the following scenarios. We have the original relational dataset that consist of seven tables. These contain information related to the patient, his or her address and doctor, as well the encounter in which they were referred to a laboratory test. The personal patient information in these datasets was anonymized to protect the patient's privacy. Table 1 presents information about tables and columns, as well as the number of records in each table. For our experiments, we have made up two laboratories, which operate separately with their own RDBMS. We distributed the datasets stored in laborder and labresult into the databases of two laboratories, which are NS lab and ON lab with schema heterogeneity as presented in table 2.

TABLE 1. LABORATORY RELATIONAL DATABASE

Table Name	Column Names	Nr. of Records
address	PRIMARY DR PMB, DR PBM IN MILLENIUM, STREET ADDR, STREET ADDR2, STREET ADDR3, PROVINCE, POSTALCODE	1589
doctor	DR LASTNAME, DR FIRSTNAME, PRIMARY DR PMB	1404
encounter	SCRAMBLED ENCOUNTER ID, SCRAMBLED PATIENT ID, PRIMARY DR PMB, E ENCTR CLASS DISP	99900
laborder	ORDER ID, SCRAMBLED ENCOUNTER ID, ORDER STATUS, CANCEL REASON, ORDER DATE, ORDER TIME, REG DT TM, ORDERABLE, VE PARENT EVENT SET DSP	582817
labresult	ORDER ID, TASK ASSAY CD, RESULT ID, RESULT STATUS, PERFORM RESULT ID, RESULT DATE, RESULT TIME, RESULT TYPE, NORMAL DISP, NORMAL ALPHA, RESULT VALUE ALPHA, NORMAL HIGH, NORMAL LOW, NUMERIC RAW VALUE, RESULT VALUE NUMERIC, ASCII TEXT	15424773
patient	SCRAMBLED PATIENT ID, PATIENT AGE, PATIENT BIRTH DATE, PATIENT GENDER	204626
task	DETAILED TASK, TASK ASSAY CD	41

TABLE 2. TABLE DESCRIPTION FOR SOURCE DATABASE AND TARGET DATABASE

DB0: nslab, NS Lab		DB1:onlab, ON Lab		DB2:integratedhealth, Integrated Health DB	
Table	Columns	Table	Columns	Table	Columns
laborder	ORDER_ID	onlaborder	ID	integratedlaborder	LAB_ID
	ORDER_STATUS		ORDER_ STATUS		REQUEST_ TIMESTAMP
	CANCEL_REASO N		REQUEST_ TIMESTAMP		TEST_TYPE
	ORDER_DATE		LAB_TYPE		CREATE_ DATE
	ORDER_TIME	orderresult	ORDER_ RESULT_ID	integratedresult	RESULT_ID
	ORDERABLE		ID		LAB_ID
labresult	RESULT_ID		RESULT_ STATUS		RESULT_ TIMESTAMP
	ORDER_ID		RESULT_ TIMESTAMP		RESULT
	RESULT_STATUS		RESULT_ TYPE		CREATE_ DATE
	RESULT_DATE		RESULT		
	RESULT_TIME				
	RESULT_TYPE				
	RESULT_VALUE_ NUMERIC				

We conducted data integration with data standardization from these two source databases to the target database, which was called integratedhealth. The application was able to handle data merging from two columns in the source database to the target database. For example, two columns, which are ORDER_DATE and ORDER_TIME in a laborder table of nslab database, were merged into a REQUEST_TIMESTAMP column in a column in an integratedlaborder table of integratedhealth database. In terms of data validation after data integration, the application was able to convert local medical terms to SNOMED Clinical Terms. For example, 'Urea' was used in the ORDERABLE column in a laborder table of nslab database and 'Urea cream' was used in the standard term database, 'Urea cream' was uploaded into a TEST_TYPE column in an integratedlaborder table of the integratedhealth database instead of 'Urea'. Since data integration was performed and generated two integrated tables located in HBase, which are integratedlaborder and integratedresult, the application could be able to handle data selection based on these integrated tables and generated a table with selected columns by a Hive query language process. However, there was noticeable time taken to generate a table due to the batch processing nature of HQL. For the EDA subsystem, the application was able to handle data selection using RImpala and stored the result dataset in the Shiny framework. Also, it provided data visualization based on R analysis. For example, we conducted an RImpala query to obtain the total number of patients, the total number of male patients and the total number of female patients for each orderable and saved the result to dataset in the Shiny framework.

The application was able to generate graphical charts such as scatter plots and line graphs and some analysis result of R function. For the EDA subsystem, we conducted a Hive query language process to get maximum processing time and average processing time in NS lab for each orderable and orderable distribution and stored the result in HBase to provide a graphic chart.

As we mentioned before, we conducted the non-functional testing such as comparing the execution time for SQL join query statement and the result of the testing is shown in table 3. We intended to compare application performance with HDFS/MapReduce and without HDFS/MapReduce by comparing execution time for SQL join query statement on MySQL and Hive Store. Since the result of Hive query was stored in HBase for web services, the execution time spent for Hive query was just reference to provide approximate time taken for Hive query process on HDFS/MapReduce. It was clear it would take much longer time for user to get responses if we use RDBMS such as MySQL. For example, the response time would take more than 24.047 seconds if there were SQL query statement with more than 3 joins to be executed for a user request. However, the response time would be around 0.055 seconds on HBase because de-normalisation is main characteristic on HBase. Also, we could compare application development efficiency between DA and EDA subsystems by comparing execution time for SQL join query statement on Hive Store and RImpala.

TABLE 3. EXECUTION TIME FOR SQL JOIN QUERY STATEMENT (SEC)

SQL	SQL with 1 Join	SQL with 2 Join	SQL with 3 Join
MySQL	0.958	2.453	24.047
Hive Store	8.30	30.180	72.460
RImpala	32.45	34.48	40.49

VI. RELATED WORK

Since there is a lack of related work to show how to build a health data analytics platform, the main novelty of our approach is that we present the entire development phrase of building a health data analytics platform considering all aspects from data integration to data visualization. There are many different approaches for data integration. However, we can categorize them into two main approaches such as data warehousing approach, which integrates by bringing the data into a single physical warehouse [9,10] and virtual data integration, which leaves the data at the sources and access it at query time [4-8]. Also, there are some other approaches by storing accumulated query results [11] and record integration [12]. Since we need data standardization before data integration and data validation after data integration, we used data warehousing approach with semi-automatic schema mapping. Although there are many approaches for data interoperability among health information systems [16-19], most of them deal with seamless data exchange between medical institutions or hospitals using HL7. In our case, we dealt with data interoperability for medical data itself. Therefore, we used SNOMED CT as a standard medical vocabulary [14,15] because SNOMED CT is a globally adopted standard health care terminology as mentioned in the previous section.

VII. CONCLUDING REMARKS

Our proposed system performs data standardization, data integration, data selection, data analysis and data visualization as explained in the previous sections. However, we do not provide data analysis and data visualization by selecting related attributes on the result tables at present for the DA subsystem since those functions are still under development. With respect to data integration, the most import task is a schema design in the integrated DB in HBase for data analysis and data visualization. Since HBase is a column oriented database with distributed system [26], it is well fitted into medical data because medical data is usually stored in some particular attributes. Also, the system could provide quick access to health data in the integrated database in HBase for

data analysis and data visualization because the rowkey composed of a column key and some other values in HBase is already indexed. With respect to data standardization, the system performs excellently with our lab data. However, the system needs to have the ability to handle health data composed of health related terms and non-health related terms. With respect to data selection, it is difficult to generate the result table with selected data if there are more than three common attributes between tables. Therefore, the schema design in HBase should be carefully designed with deep consideration. Also, the system needs to provide approximation time taken for both data selection and data analysis because Hadoop/MapReduce is batch processing. Finally, the system needs to handle various algorithms for data analysis and provide various kinds of graphical data visualization to users.

ACKNOWLEDGEMENTS

This research is supported by a discovery grant funded by National Science and Engineering Research Council (NSERC), Canada.

REFERENCES

[1] A. Podgorelec, B. Grasic and L. Pavlic. (2009, Aug.). Medical diagnostic process optimization through the semantic integration of data resources. Computer methods and programs in biomedicine. 95(2), pp. S55-S67.

[2] A. Ryan and P. Eklund, "A framework for semantic interoperability in healthcare: A service oriented architecture based on health informatics standards," in Studies in health technology and informatics, vol. 136, eHealth Beyond the Horizon, 2008, pp. 759-764

[3] IHTSDO, History: SNOMED, http://www.ihtsdo.org/about-us/history/snomed/, 2007. Last accessed: 22/05/2014.

[4] Yunmei Shi, Xuhong Liu, Yabin Xu and Zhenyan Ji. (2010). Semantic-based data integration model applied to heterogeneous medical information system. Presented at Computer and Automation Engineering (ICCAE).

[5] J. Ge, X. Xu, Y. Huang and M. You. "Semantic integration framework based on domain ontology construction," in Proceedings of the international conference on information Engineering and Application (IEA), 2012, pp. 391-399

[6] M. Mahoui, H. Kulkarni, N. Li, Z. Ben-Miled and K. Borner. "Semantic correspondence in federated life science data integration systems," in Data integration in the life science, 2005, vol. 3615, pp. 137-144

[7] C. A. Yaguinuma, G. F. Afonso, V. Ferraz, S. Borges and M. T. P. Santos. (2010). A fuzzy ontology-based semantic data integration system. Presented at Information Reuse and Integration (IRI), 2010 IEEE International Conference On. 2010.

[8] A. Doan and A. Y. Halevy. Semantic integration research in the database community: A brief survey. *AI Magazine 26(1),* 2005, pp. 83.

[9] A. Ardestani, H. Nemati, O. Eleti and F. Sadri. RxSem. (2012). A rule based semantic integration method for medical informatics. Presented at Information Reuse and Integration (IRI), 2012 IEEE 13th International Conference On. 2012.

[10] D. Moner, J. A. Maldonado, D. Bosca, J. T. Fernandez, C. Angulo, P. Crespo, P. J. Vivancos and M. Robles. (2006). Archetype-based semantic integration and standardization of clinical data. Presented at Engineering in Medicine and Biology Society, 2006. EMBS '06. 28th Annual International Conference of the IEEE.

[11] V. Podgorelec, B. Grašič and L. Pavlič. Medical diagnostic process optimization through the semantic integration of data resources. *Comput. Methods Programs Biomed. 95(2, Supplement),* 2009, pp. S55-S67.

[12] Chi Po Cheong, C. Chatwin and R. Young. (2009). A RDF-based semantic schema mapping transformation system for localized data integration. Presented at Anti-Counterfeiting, Security, and Identification in Communication, 2009. ASID 2009. 3rd International Conference On.

[13] W. Raghupathi and V. Raghupathi, "Big data analytics in healthcare: promise and potential, Big data in medicine and health, vol. 2, [Online]. Available: http://www.hissjournal.com/content/2/1/3

[14] R. J. Kate. Towards converting clinical phrases into SNOMED CT expressions. *Biomedical Informatics Insights 6(Suppl 1),* pp. 29. 2013.

[15] J. Patrick, Y. Wang and P. Budd. (2007) An automated system for conversion of clinical notes into SNOMED clinical terminology. Presented at Proceedings of the Fifth Australasian Symposium on ACSW Frontiers-Volume 68.

[16] A. Ryan. Towards semantic interoperability in healthcare: Ontology mapping from SNOMED-CT to HL7 version 3. Presented at Proceedings of the Second Australasian Workshop on Advances in Ontologies - Volume 72. 2006, [Online]. Available: http://dl.acm.org/citation.cfm?id=1273659.1273668.

[17] D. M. Lopez and B. Blobel. Enhanced semantic interoperability by profiling health informatics standards. *Methods Inf. Med. 48(2),* pp. 170. 2009.

[18] R. H. Dolin and L. Alschuler. Approaching semantic interoperability in health level seven. *J. Am. Med. Inform. Assoc. 18(1),* pp. 99-103. 2011.

[19] W. A. Khan, A. M. Khattak, M. Hussain, M. B. Amin, M. Afzal, C. Nugent, S. Lee, An adaptive semantic based mediation system for data interoperability among health information systems. Journal of medical system, 38(28), June 2014.

[20] G. Mecca, P. Papotti, D. Santoro, G. Mecca, P. Papotti and D. Santoro. "A short history of schema mapping systems," in Proceedings of SEBE 12, 2012, pp. 99-106

[21] G. Liu, S. Huang and Y. Cheng. "Research on semantic integration across heterogeneous data sources in grid," in Frontiers in computer education, 2012, Available: http://dx.doi.org/10.1007/978-3-642-27552-4_56

[22] D. H. Lee, F. Y. Lau and H. Quan. A method for encoding clinical datasets with SNOMED CT. *BMC Medical Informatics and Decision Making 10(1),* pp. 53. 2010.

[23] Language Manual, https://cwiki.apache.org/confluence/display/Hive/LanguageManual, 2014. Last accessed: 05/08/2014

[24] Hadoop Hive, http://archive.cloudera.com/cdh/3/hive/language_manual/joins.html, 2007. Last accessed: 06/08/2014

[25] Apache Mahout, https://mahout.apache.org/, Last accessed: 07/08/2014

[26] Apache HBase, http://hbase.apache.org/, Last accessed: 08/08/2014

[27] R. Fagin, L. M. Haas, M Hernaudez, R, J. Miller, L. Popa and Y. Velegrakis. "Clio: Schema mapping creation and data exchange," in lecture notes in computer science, Vol. 5600, pp. 198-236, 2009

[28] D. Maier, A. O. Mendelzon, and Y. Sagiv. Testing implications of data dependencies. ACM Transactions on Database Systems (TODS), 4(4):455-469, 1979

[29] C. Beeri and M. Y. Vardi, A proof procedure for data dependencies, J. ACM, 31(4), pp. 718-741, 1984

[30] Google Charts, https://developers.google.com/chart/, Last accessed: 10/02/2014

[31] Shiny by RStudio, http://shiny.rstudio.com/, Last accessed: 10/02/2014

[32] The R project for statistical computing, http://www.r-project.org/, Last accessed: 10/02/2014

[33] The R project for statistical computing, http://www.r-project.org/, Last accessed: 10/02/2014

[34] Apache Hive, http://hive.apache.org/, Last accessed: 10/02/2014

[35] Impala, http://impala.io/, Last accessed: 10/07/2014

[36] Java Database Connectivity, http://www.oracle.com/technetwork/java/javase/jdbc/index.html, Last accessed: 10/07/2014

[37] RImpala package, http://www.inside-r.org/node/223456, Last accessed: 10/07/2014

[38] XML, http://www.w3.org/XML/, Last accessed: 10/07/2014

[39] MySQL, http://www.mysql.com/, Last accessed: 10/07/2014

[40] RImpala: R and Impala, http://cran.r-project.org/web/packages/RImpala/index.html, Last accessed: 10/08/2014