International Journal of Multidisciplinary Research And Studies

Available online at https://ijmras.com/

INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH AND STUDIES

ISSN: 2640 7272

Volume:03; Issue:01 (2020)

METADATA ADMINISTRATION FOR HEALTH CARE DATA AMALGAMATION

MD. ANWAR

M.Phil., Roll No.: 150132 Session-2015-16
Department of Computer Science, B.R.A. Bihar University, Muzaffarpur, India
md.anwarali87@gmail.com

Abstract

The lack of fine-grained, cross-cohort query, and exploration interfaces and sys- tems. Although many data repositories allow users to browse their content, few of them support fine-grained, cross-cohort query, and exploration at thestudy-subject level. To understand the challenge, we provide a review of thekey concepts. In clinical research, investigators tend to work independently or in clusters of researchteams. Raw data collected from experiments or clinical trials are usually stored elec-tronically on a computer. However, to perform independent analysis or verify ex- perimental results, sharing data between different researchers or teams is necessary. Furthermore, sharing and reuse of data is important for facilitating scientific discoveryand enhancing research reproducibility.

keywords: Metadata, Health, Amalgamation

INTRODUCTION

Patient data are growing at an explosive rate in the medical field with the wide adop-tion of electronic health records (EHR). Patient data cover patient demographics, diagnosis, laboratory tests, medications, images, and genome sequences. With a large amount of clinical data integrated, efficient data retrieval and exploration have be- come a challenging issue. Specific challenges include:

- Barriers between data exploration and research hypotheses. In a traditional clinical research workflow, research hypotheses come before patient data acqui- sition. If the research hypotheses and acquired patient data do not support thehypotheses, then the study design needs to be adjusted. A new and efficient data exploration tool is needed to accelerate the process. With such a tool, re-searchers can explore the data to provide preliminary evidence to their research hypotheses before the start of a clinical trial.

- The lack of fine-grained, cross-cohort query, and exploration interfaces and sys- tems. Although many data repositories allow users to browse their content, few of them support fine-grained, cross-cohort query, and exploration at thestudy-subject level. To understand the challenge, we provide a review of thekey concepts.

– Fine-grained. A fine-grained query is a highly-customizable query with low granularity and high details.

– Cross-cohort. A cohort study is a particular form of a longitudinal study that samples a cohort through time. A cross-cohort query means to queryand fetch data from multiple cohort studies at the same time.

– Study-subject. The United States Department of Health and Human Ser- vices (HHS) defines a human study subject as a living individual about whom a research investigator obtains data through 1) intervention or in- teraction with the individual, or 2) identifiable private information. Exploration at the study-subject level is the result of a fine-grained query.

To find a male patient with asthma under 50 years old, a typical SQL statement is SELECT * FROM patients WHERE gender = 0 AND asthma = 0 AND age

¡= 50. From the perspective of end-users, an interface with SQL like query capability can help their data exploration capability.

Fine-grained Data Exploration of Heterogeneous Datasets

In clinical research, investigators tend to work independently or in clusters of researchteams. Raw data collected from experiments or clinical trials are usually stored elec-tronically on a computer. However, to perform independent analysis or verify ex- perimental results, sharing data between different researchers or teams is necessary. Furthermore, sharing and reuse of data is important for facilitating scientific discovery and enhancing research reproducibility. Multiple data repositories have been built and are accessible to researchers, such as GDC - the National Cancer Institute's Genomic Data Commons BioPortal - a repository of biomedical ontologies OpenfMRI - a repository for sharing task-based fMRI data and NSRR - the National Sleep Research Resource. These data repositories allow an investi- gator to browse and download data under certain restrictions. However, not many of them can enable users to conduct fine-grained, cross-dataset query, and explore of the study-subject level before users decide which dataset to gain further access. Study-subject level exploration can help researchers to quickly assess the feasibility ofstudies or verify the research hypothesis without requesting further access and avoid unnecessary data analysis. Researchers will be able to have a sense of the dataset without downloading the whole dataset.

CONTRIBUTIONS

To overcome these gaps and challenges, we propose a general framework called Meta- Sphere. MetaSphere provides three major functionalities in terms of metadata man- agement for clinical data integration. The first functionality is the structural, scalable, and computer understandable way of metadata storage. MetaSphere stores the on- tology and its associated concepts, variables, and domains in a scalable database. Additionally, utilizing the database's associations between tables, MetaSphere can represent the relationships between concepts, the relationships between concepts and variables, the relationships between variables and domains properly.

The second functionality is the fine-grained, cross-cohort query interface. MetaS- phere hierarchically organizes ontology and its concepts and reflects such hierarchies in the interface. With direct interaction, users will be able to browse the ontology's structures easily. Utilizing the query interface, users can compose complex queries to query and explore data at the study-subject level.

Finally, MetaSphere provides an interactive, intuitive, and collaborative mapping interface for building mapping between data dictionary to ontology, so as to facilitate data analytics through interoperability and integration and provide semantic access across aggregated data used in knowledge-based applications and services.

RESEARCH METHODOLOGY

Agile methods of software development have been widely leveraged in recent years 43]. Iterative and incremental development, evolving since the 1950s, has taken the place of the waterfall model as the main-stream style of software development [42].In this chapter, we will discuss the detailed design and methodology for developing MetaSphere using agile development.

System Architecture

Figure shows the overall system architecture of MetaSphere. There are three ma-jor components: 1) Frontend query interface; 2) Backend application server; 3) Databases. These components are loosely decoupled but seamlessly combined as a functional application.

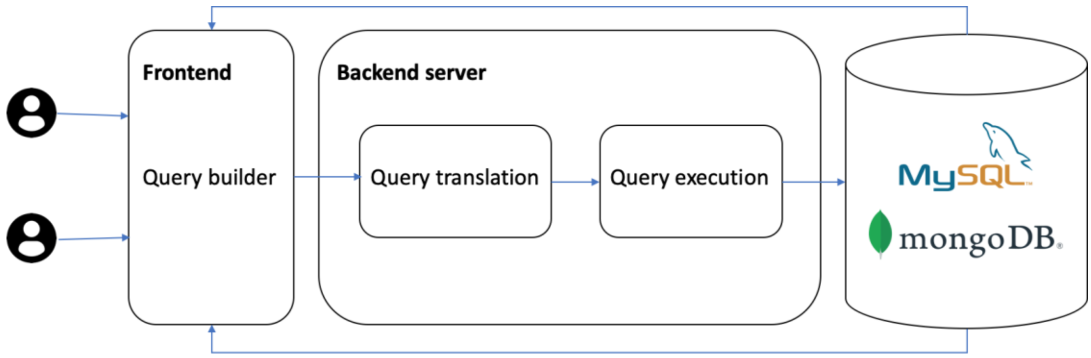

Figure 1 System Architecture Overview

ReactJS - A JavaScript Library

ReactJS is a JavaScript library for building user interfaces. It is created and maintained by Facebook. It is used as a base in developing high-performance single- page applications. ReactJS has become one of the widely used frameworks for building frontend interfaces. There are several features which make it extremely successful and these features perfectly match our development requirements.

Components based. The design philosophy of ReactJS is to separate a web interface into different components. A root component is the entry point of the interface. Each component has its own children's components. In such a way,an interface becomes a tree. Moreover, every component can be reusable since its a placeholder to render different data. A typical interface will have many repeated elements, such as many rows in one table. We then can make a rowas an individual component and pass in different data. ReactJS enhances the reusability of codes even for frontend interface coding.

Virtual dom. Another notable feature is the use of a virtual Document Object Model or virtual DOM. React creates an in-memory data structure cache, com- putes the resulting differences, and then updates the browser's displayed DOM efficiently . As shown in Figure 2, this allows the programmer to writecode as if the entire page is rendered on each change, while the React libraries only render subcomponents that actually change. The virtual DOM feature makes ReactJS updates efficient.

Single direction data flow. The data flows from the components itself to its children components. With such setting, developers will be able to catch unex- pected bugs quickly and easily. Figure 3 demonstrate the data flow in ReactJS.

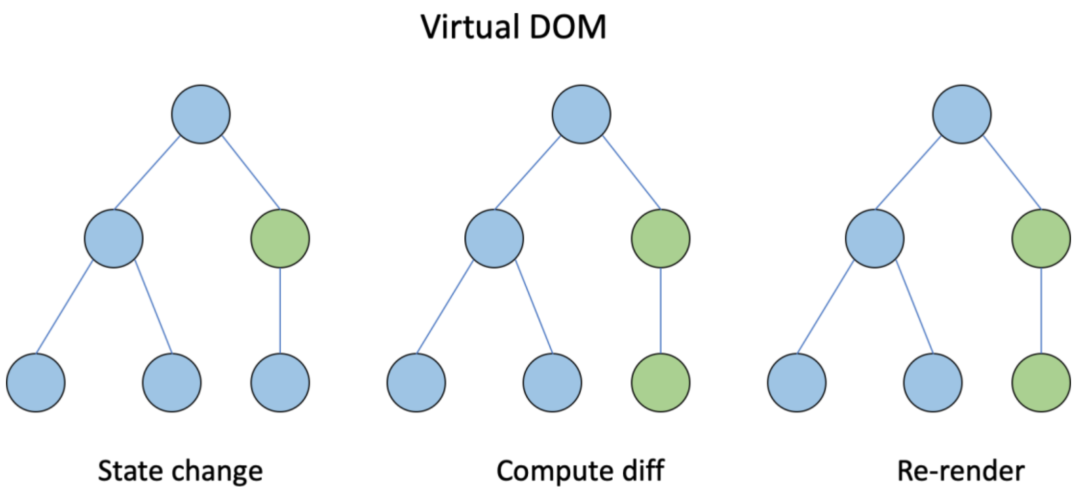

Figure 2 Virtual DOM.

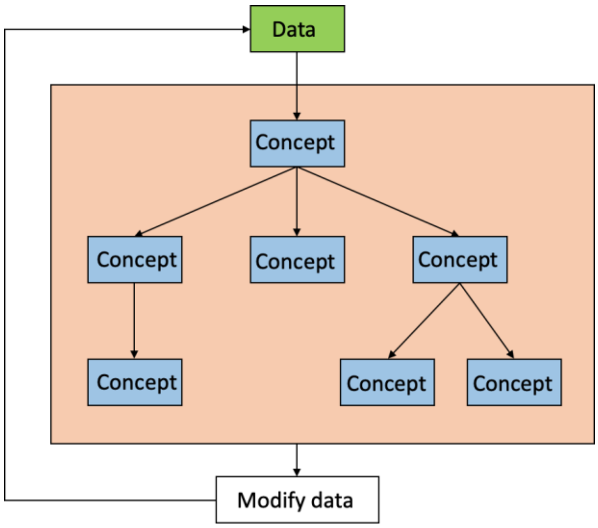

Figure 3: One direction data flow.

The aforementioned features make React JS a decent choice to build our MetaS- phere frontend interface. Especially, we would like to represent the ontology hierarchi- cal tree structure. In another way, we could view the component as a typical class ina programming language and we are turning the interface design into object-oriented programming. Figure 3.4 shows the detail of the core design. There are also other components but the major components are QueryDashboard, ConceptList, Concept,ConceptWidget. Numerical and Categorical components are the two most common types for a Concept Widget component.

Query Dashboard. The Query Dashboard component is the root component for the query interface and its the entry point of our interface. Most of the uses would spend their visit in this component. When the user performs a query, Query Dashboard will gather all the QueryWidget information and send out a request to the backend server to perform a query.

ConceptList. The ConceptList component is a functional component. It is the component that fetches data from the backend server and handles all the logicrelated to concept display. Concept. The Concept component is called a representational component or render component. The only responsibility for the Concept a component is to render actual concept data in the interface.

Concept Widget. The Concept Widget component is a visual representation ofa specific concept type. The Concept Widget component will render different child components based on the passed in concept type.

Numerical. The Numerical component is a QueryWidget. It is the correspond- ing component for a numerical concept. It contains a slider bar for users to perform a range-based query, which would produce a minimal and maximum value for the concept.

Categorical. The Categorical component is also a QueryWidget and it is related to categorical concepts. It will render all the domains(options) for users to select. For instance, a gender concept will have options male and female.

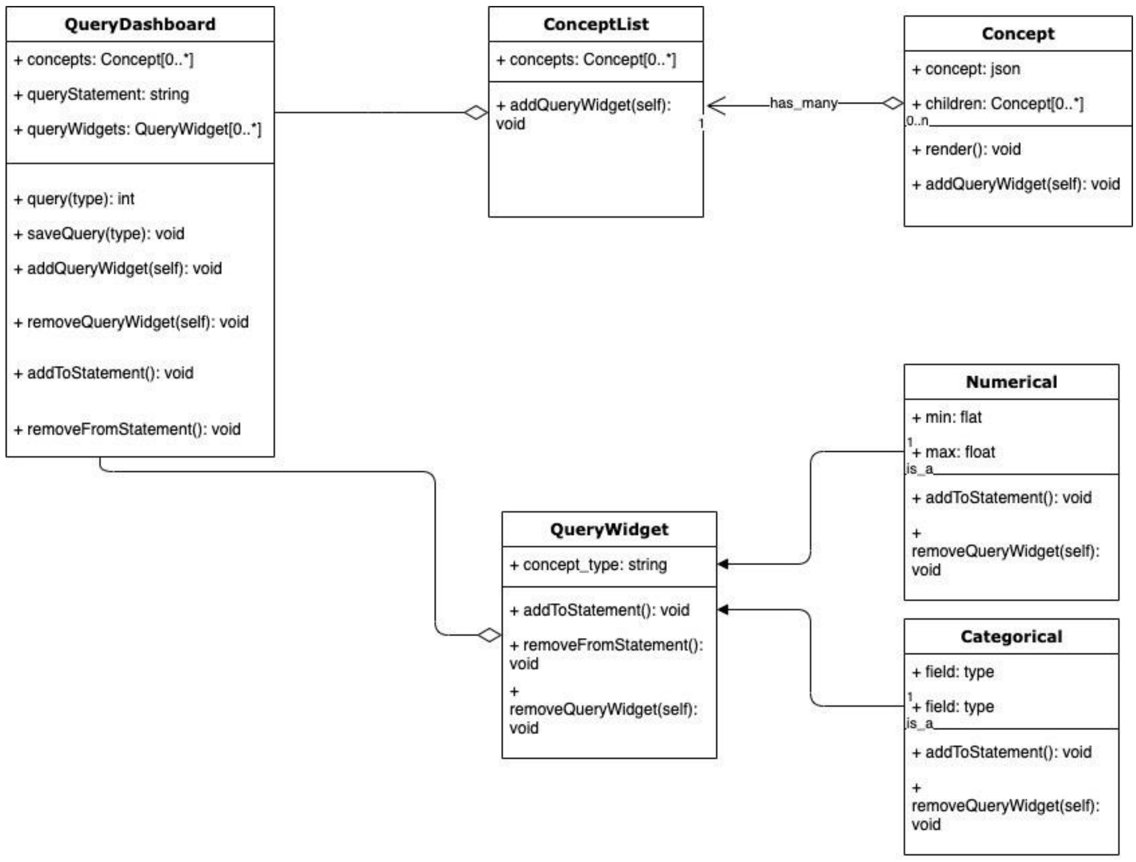

Figure 4 UML diagram of the frontend query interface

RESULT

Data repository

We used MySQL databases to store the nine datasets. Table 4.3 lists the names of the datasets, the names of the visits, the numbers of data elements (or variables), the numbers of subjects, and the numbers of mapped variables to the canonical datadictionary. Note that the mapped variables in each visit of a dataset are a subset of all the variables in the visit. The canonical data dictionary contained a total of 919 common data elements (554 of them are specific to the sleep research domain and 365 of them are common across study domains). Among them, 42 were detected to have inconsistent codings across different datasets, including ”gender,” ”race,” ”history of asthma,” and ”history of sleep apnea.” A total of 830 mappings from heterogeneous codings to the uniform codings were created to harmonize the data with inconsistent codings. In addition, 57 elements in the canonical data dictionary were linked to the NIH Common Data Element (CDE).

Cross-cohort exploration engine

We implemented the X-search cross-cohort exploration engine using Ruby on Rails, an agile web development framework. It has been deployed at https://www.x-search.net/

Table 1 Summary information for each of the nine datasets.

Dataset	Visit(s)	No. of variables	No. of subjects	No. of mapped variables
SHHS	shhs1	1266	5804	615
	shhs2	1302	4080	592
CHAT	baseline	2897	464	826
	followup	2897	453	823
HeartBEAT	baseline	859	318	158
	followup	731	301	103
CFS	visit5	2871	735	1023
SOF	visit8	1114	461	350
MrOS	visit1	479	2911	261
	visit2	507	2911	222
CCSHS	trec	143	517	94

HCHS	sol	404	16,415	97
	sueno	505	2252	5
MESA	sleep	723	2237	512

and open to public access for free.

Figure 5 shows the query builder interface with the four areas annotated. In the area to select datasets, all the nine datasets are chosen - five of them can be directly seen, and the other four can be seen when scrolling down. The area to construct queries contains two query widgets for ”gender” (with checkboxes) and ”age” (witha slider bar), with specified query criteria: female, and age between 20 and 50. Thearea for query results shows the numbers of subject counts meeting the query criteriain each dataset, as well as the total number of subject counts.

Figure 6 gives an example of the graphical exploration interface, where the term for the y-axis is specified as ”body mass index” and the term for the x-axis is ”history of diabetes”. The box plot shown in the figure is generated based on two variables in the CFS dataset mapped to ”body mass index” and ”history of diabetes” respectively and indicates that the median body mass index of patients who had a history of diabetes is greater than that of patients who had no history of diabetes.

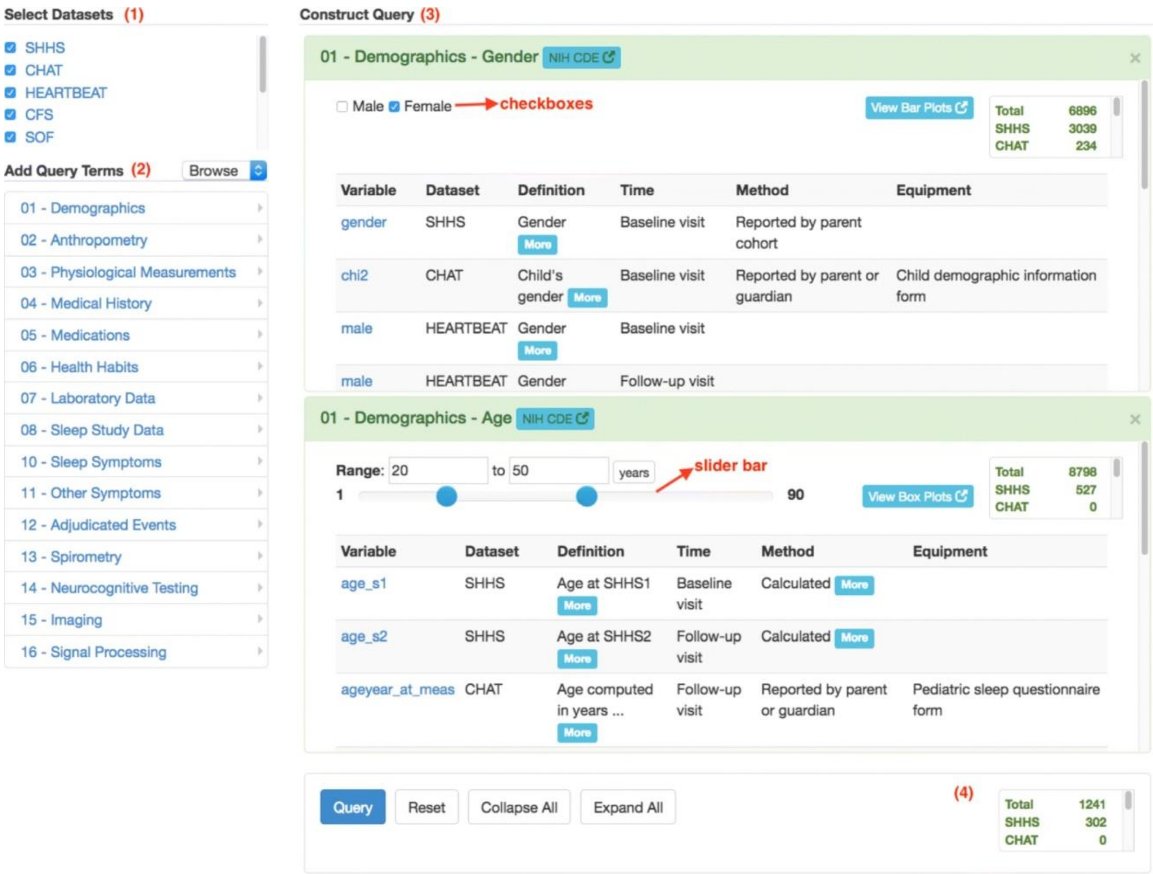

Figure 5 shows the case-control exploration interface illustrating the exemplar

Figure 5: Screenshot of the query builder interface. Four areas: (1) Select Datasets; (2) Add Query Terms; (3) Construct Query; (4) Query Results. This example queries the numbers of female patient subjects aged between 20 and 50. steps mentioned in the Methods section. This example is to explore: In elderly (base query: age between 45 and 85 years), obese people (base query: body mass index between 30 and 85) without cardiovascular disease (base query: no history of cardiovascular disease), whether the presence of self-reported diabetes (case condition: had a history of diabetes, control condition: no history of diabetes) is related to sleep apnea (outcome term: obstructive sleep apneas/hours).

The cross-cohort exploration system supports additional functionalities, including the query manager, case-control manager, and International Classification of Sleep Disorders (ICSD) query builder. Query and case-control managers allow users to save queries and case-control explorations for reuse. ICSD query builder is a dedicated query builder for more complicated ICSD terms.

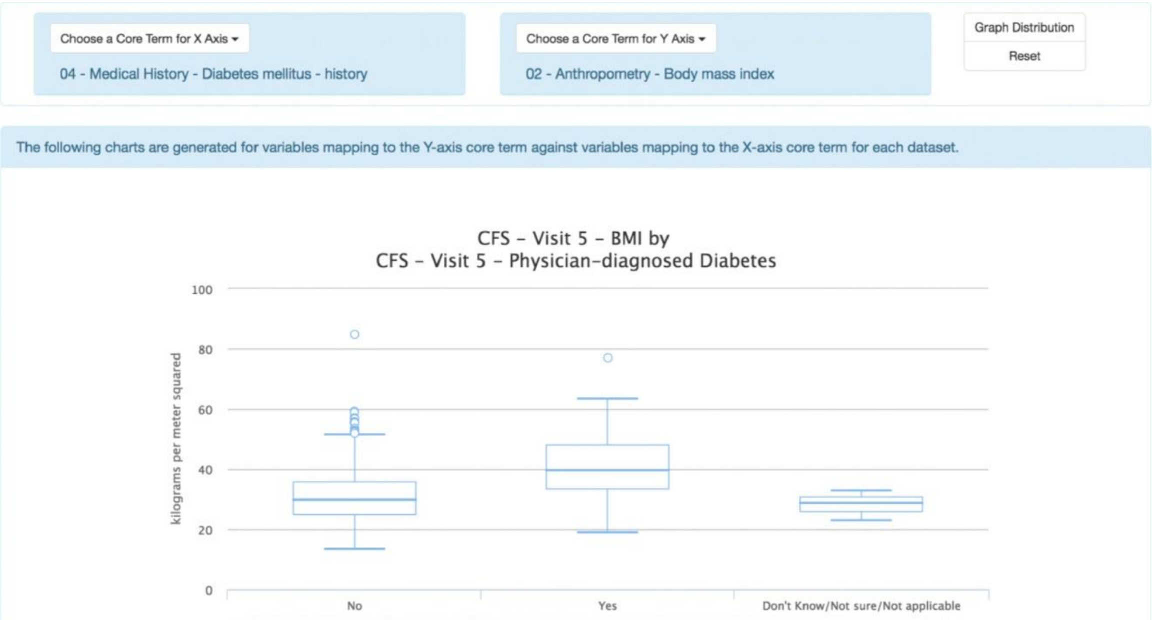

Figure 6 Screenshot of the graphical exploration interface. This example shows one of the box plots generated for body mass index (BMI) against diabetes.

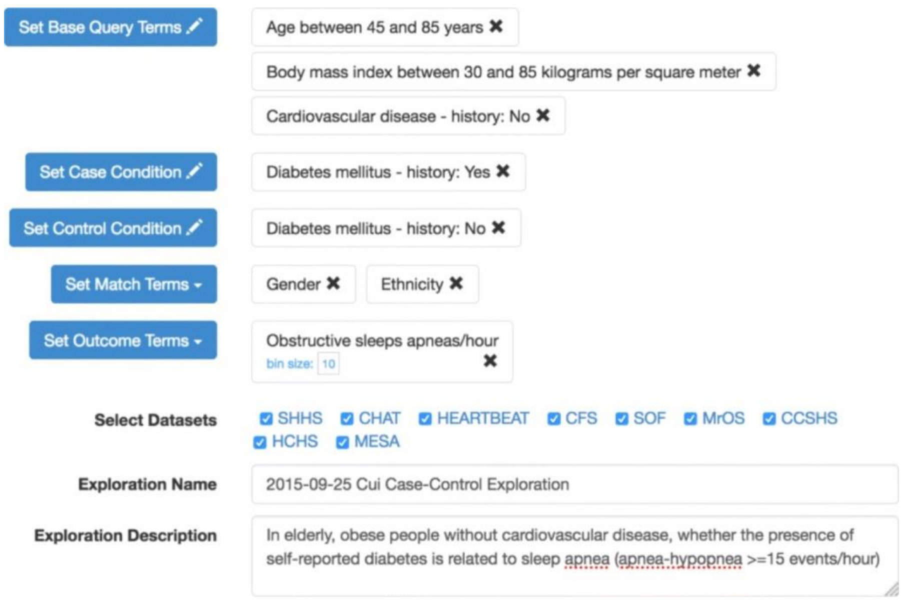

Figure 7 Screenshot of the case-control exploration interface. This example is to explore: In elderly, obese people without cardiovascular disease, whether the presence of self-reported diabetes is related to sleep apnea (apnea-hypopnea ¿=15 events/hour).

Usage

the cross-cohort exploration system has received 1,835 queries from users in a wide range of geographical regions (16 countries), including Australia, Canada, China, France, India, South Africa, the United Kingdom, and the United States. Figure 4.5 shows the number of times each of the nine datasets got queried (notethat each user query may involve multiple datasets). And the top ten query terms are: ”age,” ”obstructive sleep apneas/hour,” ”central sleep apneas/hour,” ”gender,””body mass index,” ”diabetes mellitus - history,” ”cardiovascular disease - history,” ”apnea hypopnea index greater than or equal to 15,” ”apnea hypopnea index,” and ”race.”

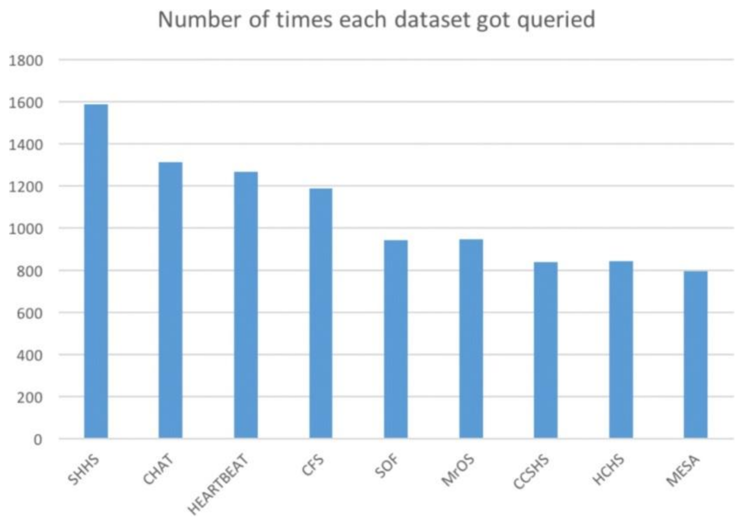

Figure 8 Numbers of times each dataset got queried.

CONCLUSION

While developing X-search, we found out that some query performance issues are introduced by the traditional relational databases. Such query performance issues can be improved but not solved completely. To address that, we tried out the NoSQL databases and conduct a comparison experiment. We developed two NoSQL-based patient cohort identification systems, in comparison to a SQL-based system, to evalu- ate their performance on supporting high-dimensional and heterogeneous data sources in NSRR. Utilizing NoSQL databases, we overcame the limitation of maximum ta- ble column count in traditional relational databases. We successfully integrated eight NSRR cross-cohort datasets into NoSQL databases, which largely enhanced the query performance compared to the MySQL-based system, while maintained similar perfor- mance for data loading and harmonization. This study indicates that NoSQL-based systems offer a promising approach for developing patient cohort query systems across heterogeneous data sources in our case.

REFERENCES

1. Tracy D Gunter and Nicolas P Terry. The emergence of national electronic health record architectures in the united states and australia: models, costs, and questions. Journal of medical Internet research, 7(1):e3, 2005.

2. What is human subjects research?. https://web.archive.org/web/20120207032034/http://www.utexas.edu/research/rsc/humansubjects/ whatis.html (visited: 2020-03-10).

3. Anca Vaduva and Thomas Vetterli. Metadata management for data warehous- ing: An overview. International Journal of Cooperative Information Systems, 10(03):273–298, 2001.

4. Francis S Collins and Lawrence A Tabak. Policy: Nih plans to enhance repro- ducibility. Nature, 505(7485):612–613, 2014.

5. Joseph S Ross and Harlan M Krumholz. Ushering in a new era of open science through data sharing: the wall must come down. Jama, 309(13):1355–1356, 2013.

6. Lisa M Federer, Ya-Ling Lu, Douglas J Joubert, Judith Welsh, and Barbara Brandys. Biomedical data sharing and reuse: Attitudes and practices of clinical and scientific research staff. PloS one, 10(6), 2015.

7. Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. The fair guiding principles for scientific data management and stewardship. Scientific data, 3, 2016.

8. Nci genomic data commons, Jan 2020. https://gdc.cancer.gov/ (visited: 2020-01-30).

9. Natalya F Noy, Nigam H Shah, Patricia L Whetzel, Benjamin Dai, Michael Dorf, Nicholas Griffith, Clement Jonquet, Daniel L Rubin, Margaret-Anne Storey, Christopher G Chute, et al. Bioportal: ontologies and integrated data resources at the click of a mouse. Nucleic acids research, 37(suppl 2):W170–W173, 2009.

10. Russell A Poldrack and Krzysztof J Gorgolewski. Openfmri: Open sharing of task fmri data. NeuroImage, 144:259–261, 2017.

11. Dennis A Dean, Ary L Goldberger, Remo Mueller, Matthew Kim, Michael Rueschman, Daniel Mobley, Satya S Sahoo, Catherine P Jayapandian, Licong Cui, Michael G Morrical, et al. Scaling up scientific discovery in sleep medicine: the national sleep research resource. Sleep, 39(5):1151–1164, 2016.

12. Guo-Qiang Zhang, Licong Cui, Remo Mueller, Shiqiang Tao, Matthew Kim, Michael Rueschman, Sara Mariani, Daniel Mobley, and Susan Redline. The national sleep research resource: towards a sleep data commons. Journal of the American Medical Informatics Association, 25(10):1351–1358, 2018.