Design and Implementation of College Consumption Analysis System Based on NoSQL Database

Haishan Zheng
Information & Network Center
Xiamen University
Xiamen, China
haishion@xmu.edu.cn

Xiaolian Jiang
Information & Network Center
Xiamen University
Xiamen, China
xljiang@xmu.edu.cn

***Abstract*—This paper is aimed at mitigating the difficulty of analyzing students' personal consumption patterns by successfully coping with the large volume of student consumption data that exists. Most universities store their consumption data in relational databases; the amount of data is huge, but the databases performance only enables it to show the students consumption records from recent days. The delay when performing big data analysis in relational databases is very large. This article proposes a method by exporting data from relational database to NoSQL database, selecting the right database to balance the delay of importing and querying time and using load balancing, caching, and other tuning strategies to reduce the response time of web application, achieving a good user experience under a high concurrency environment. Using this method, students can view their consumption data analysis in seconds.**

***Index Terms*—Big Data; Consumption Records; MongoDB; NoSQL**

I. Introduction

Colleges and universities store many students' behavioral data, such as consumption records, network logs, grades, and lending records. Many schools have analyzed this data in various methods [1]. The use of big data will affect the decision-making in schools [2] and promote the development of university informatization [3–5]. He Xiuquan [6] sorted out and summed up the relevant literature results of the analysis of consumption data to assist school decision-making. In these documents, Hadoop is used more often for analysis of consumption records [7–10], web access logs [11, 12], and lending records [13–16]. Most of the documents above are for non-real-time decision analysis and cannot show the results of the analysis to end users in real time.

College students' turnover is high, and there are a huge number of graduate and enrolled students; the total number of students historical consumption records have therefore reached a high level. Since all of the original systems were based on relational databases, it is difficult to run analysis over them. At the beginning of the system design, we considered showing students only the consumption records for the last few days. Only a few tuning methods, such as database indexing, were considered. Although a current relational database can store a large amount of data, it takes a long time in to retrieve and analyze, and running of analysis will slow down the original system [17].

This research uses rice consumption records of students in Xiamen University as the data set and exports the data of the original relational database to the NoSQL distributed database, then optimizes it according to the needs of later use when exporting. By testing a variety of different NoSQL databases, the delay on importing and querying is balanced. After mapping the database file into the memory to improve the performance, the individual student's consumption records can be statistically analyzed in less than one second. By adding statistics and analysis to the original system, our application attracted a large number of students to visit and share statistical information on social networks.

Supported by Education Informatization Special Project 2016 of China Association of Higher Education (Grant No. 2016XXZD06)

II. requirements

Based on the survey of students, we break their ideas down into functionality. Students want to analyze their behaviors related to the consumption of rice, including the amount of time spent at restaurants, their favorite restaurant, and other considerations. Through the investigation of some students, this system summarizes some of the most common feature needs of students, Fig. 1 shows the use cases of the system.

- Students can view how much rice they consumed during school.
- Students can view at what time and at which restaurant the first purchase occurred.
- Students can view the day during which the most rice is consumed.
- The histogram shows the total amount of students' spending in each restaurant.
- The pie chart shows the distribution of students' rice consumption.
- The histogram shows the amount of consumption in students spending each year.
- The histogram shows the student's time point on daily consumption.
- The histogram shows the amount of students' spending for on each of three meals each day, and whether or not they had breakfast.

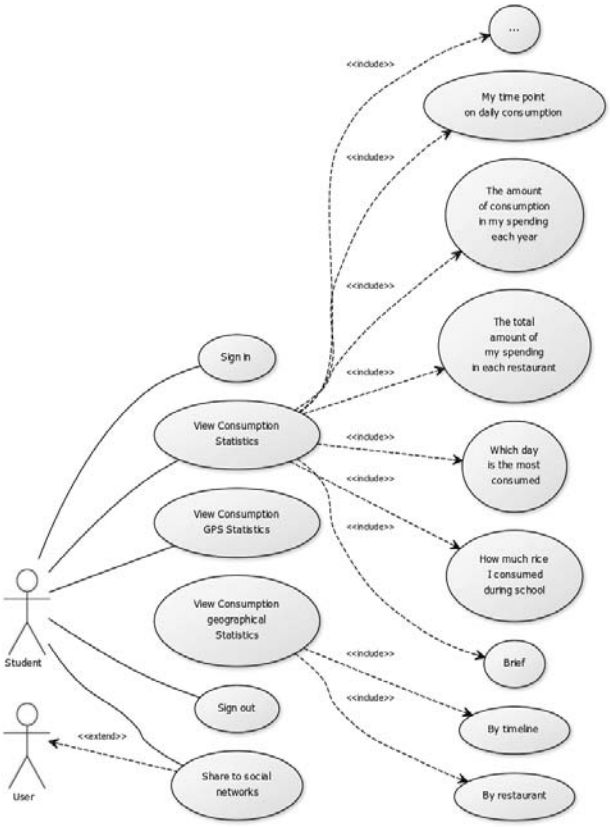

Fig. 1. Use cases of the college consumption analysis system.

- Overlapping pie charts on a map shows the proportion of students spending in restaurants, using different colors to distinguish three meals.
- The usage timeline shows the consumption footprint at each restaurant over time.
- Students can share their own consumption statistics on social networks. To protect student privacy, we limit the shared pages exposure, giving them an expiration time after which they are no longer viewable.

III. Design

The system is divided into two parts. The first part deals with the raw data of the Oracle production database and saves it into a new database for web system calls. We write Python scripts to grab data from the Oracle production database and store it in the MongoDB database [18]. The script is automatically run through the scheduled task, and newly generated data is regularly captured every day.

The second part is the web application, which is accessible to all students and internet users. After sharing via social networks, the number of visits to the website will accelerate in growth. To support high concurrency and reduce response time, the system has designed the following strategies, Fig. 2 is the system framework diagram.

- We use Nginx as the load balancing server. After the user's request arrives at the load balancing server, it distributes the request to multiple web servers.

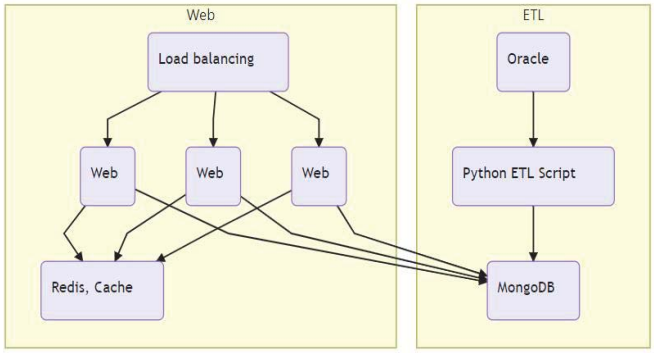

Fig. 2. System framework of the college consumption analysis system.

- We separate the static and dynamic page content and use JavaScript to load dynamic content. Consistent page content for all users is written as a static HTML file, given directly by the load balancing server.
- For the dynamic content obtained by JavaScript, since there is a one-day delay in data capture, the data will not change within a day. Therefore, after the web server reads data from MongoDB, it caches these data for half a day. In this way, later visitors do not need to recalculate within half a day.
- Since the data shared by students is visible to all users, we can directly cache the entire web page. We chose to cache the web page for 24 hours in Nginx. After using this method, users do not need to access the backend web server and MongoDB database to get data when viewing the shared page multiple times, which relieves the pressure on the backend servers.
- Caching the shared page in users browsers for 4 hours reduces the interaction with the user and the load balancing server in repeated access.

Since there is GPS information in the restaurant locations of the user's consumption, the geographical information of the restaurants patronized by the user can be displayed on the map by establishing a mapping table.

IV. Data Processing

A. Problems with Production Database

- It is not possible to perform statistical analysis directly on the production database, as it will slow down the production database and affect the systems operation.
- The schema of the production database cannot be modified. The schema of the production database is only for the needs of the original system. For example, in the rice consumption data, the date and time fields on the production database are separated from each other, and the date field is stored as a string. Consequently, a query on the date field is slow.
- The production database cannot add indices. Adding an index will cause the insertion speed to slow down. The original system does not need a new index; for example, in the rice consumption data, only the date is currently indexed, the student ID is not indexed.

- The production database does not have an auto-increment ID. It unable to save progress when importing data.
- The amount of data in the production database is very large; for example, the amount of data in a single table has reached 100 million. The query speed is therefore very slow.

B. Data Preprocessing

In order to solve the problems above, we usually use the ETL (extract, transform, load) method to query the original data, transform it and then load it into a new database. As the system continues to operate, new consumption data will be generated, and extraction will need to consider how new data will be re-imported on a regular basis. The use of auto-increment ID is a common method, but there is no such ID in the database table. The legacy table has a date-based index, so we use the date-based index to save progress in our extract script. We run a script every day to gather data from the last date to yesterday. Due to the existence of the date-based index, and the query time point set in middle night, the query will not affect the operating speed of the production environment.

To standardize the fields and prepare for the analysis, the original fields are converted. The original date and time fields are stored separately using a string format. The two fields are integrated into a date format in the new database. The time string is also stored unchanged for redundant manner. The original and converted format is shown in the Table I.

C. Database Selection

After the data is loaded into the database, there are several ways to perform calculations when the user visits the statistics page:

- User queues within the system. After the user clicks the web page, his request is added to the calculation queue; the user needs to wait for the calculations of other users in the queue to obtain the results. Calculations can be performed on multiple servers in parallel to reduce waiting time.
- Calculate each users results after the data is loaded each day. This method will produce a large amount of calculation; if a user does not come up to view their own statistics eventually, it will cause an unnecessary waste of computing resources. The calculations also take time; during this time, the system may not provide other services.
- Calculate when data is extracted. Each time the data is extracted, we fetch the old result from the new database, merge with the new data, and load it into the new database. This merge method takes a while, and the overall ETL time becomes longer. As the script needs to remain connected to the production database the entire time, the production database will be affected.

All the above methods have their own shortcomings. Finally, we have achieved a short loading time through the testing, selection, and optimization of the database; this method displays results instantaneously after each inquiry.

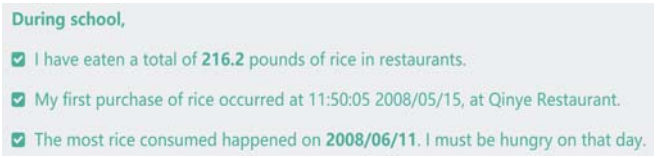

Fig. 3. Screenshot of the college consumption analysis web application.

We have tested several databases and methods to balance load and query times.

- Using Elasticsearch [19]. Elasticsearch is a highly scalable search and analytics engine. Data can be distributed in multiple Data Node servers. An index in Elasticsearch is like a table in a relational database. In the choice of the index, we chose to write data to one index by day in the beginning, because the extraction takes place daily. While the writing speed is very fast when data are all be written in one single index, the query speed is very slow because the final calculation needs to query all indices to get the results by student ID.
- Using Elasticsearch, we choose the student ID as an index. The statistical results of the final business requirements are all based on the student ID, so the query speed is very fast because it only queries one index at a time. However, since the extract script is based on date, the index needs to be switched each time when writing to different student ID indexes, and the writing speed becomes very slow.
- Use Hadoop to store the daily consumption records generated in a text file in HDFS. This method requires more servers and has a lot of delay time for query results.
- With MongoDB, since there is no transaction, and data is only written without modification, the writing speed is very fast. By compressing and mapping the data file into the memory, the query speed is very fast as well.

Finally, we chose MongoDB to save the data. Table II compares the load and query times for several different databases and methods.

V. Conclusion

The system implements the personal data statistical analysis function through the back-end data loading to MongoDB and front-end web application. Within the four hours immediately after the system launched, 10,000 users had logged in and the number of page views had reached 100,000. The response time of the web application was less than 1 second. According to our system, the students have gained a better understanding of personal consumption and related time spent. The system also addresses their own need to share their own statistics on social networks. Fig. 3, 4, 5 and 6 shows the screenshots of the web application.

This system is focused on how to retrofit the existing university legacy information systems without changing the database schema or architecture of the legacy system. We verified that it is a feasible method to export data to a new NoSQL database and perform the large-scale statistical

TABLE I
DETAIL OF DATABASE FIELDS CONVERSION OF THE COLLEGE CONSUMPTION ANALYSIS SYSTEM..

Original field name	*Student ID*	*Transaction date*	*Transaction time*	*Restaurant code*	*Transaction amount*
Original field type	VARCHAR	VARCHAR	VARCHAR	VARCHAR	NUMBER
Whether the original field is indexed	No	Indexed	No	No	No
Conversion rules	N/A	Integration from transaction date and transaction time	N/A	N/A	N/A
Field type after conversion	VARCHAR	DateTime	VARCHAR	VARCHAR	NUMBER
Whether the new field is indexed	Indexed	Indexed	No	No	No

TABLE II
THE LOAD AND QUERY TIMES FOR DIFFERENT DATABASES AND METHODS.

Database	*Method*	*One-day data loading time*	*Query delay*	*Remarks*
Elasticsearch	Use date as index	>2 hours	20 seconds	
Elasticsearch	Use student ID as index	>24 hours	<1 second	
Hadoop	Text files by date	60 seconds	10 seconds	Decrease by the number of servers
MongoDB		60 seconds	<1 second	Large memory

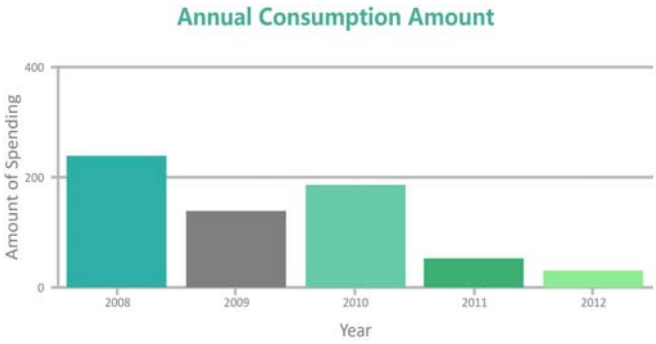

Fig. 4. Screenshot of the college consumption analysis web application.

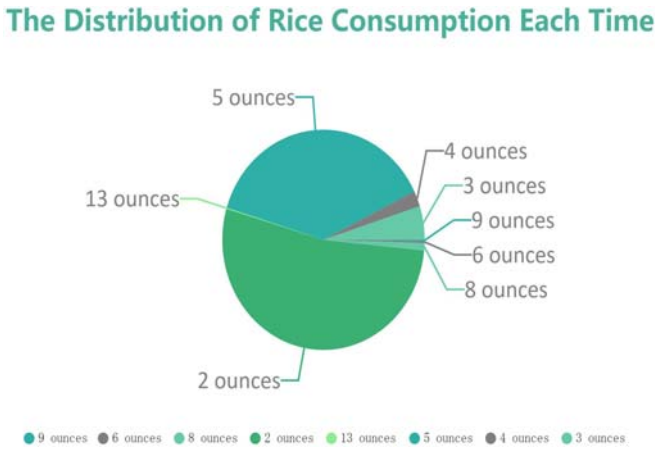

Fig. 5. Screenshot of the college consumption analysis web application.

analysis on it. This system is focused on real-time analysis. If the system does not pursue real-time analysis, using Hadoop is a relatively simple and universal method. Currently, the system only analyzes individual students' data. In the future, it can analyze the similarities of multiple students consumption habits and analyze the consumption status and busyness of individual restaurants [10]. The system analysis method and framework can also quickly access other data for analysis,

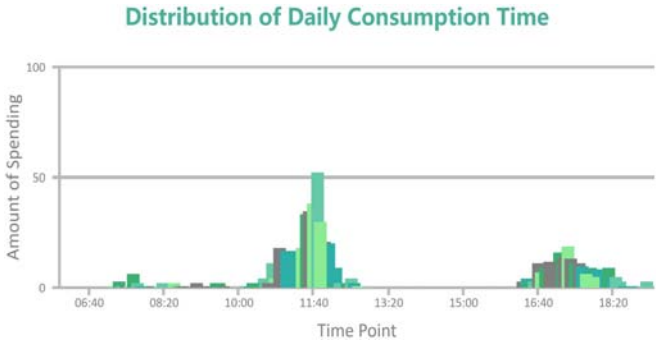

Fig. 6. Screenshot of the college consumption analysis web application.

such as library lending records, access control records, and web access logs.

REFERENCES

[1] M. Parry, "Big data on campus," *The New York Times*, vol. 18, 2012.
[2] S. John Walker, "Big data: A revolution that will transform how we live, work, and think," 2014.
[3] J. Dongxing, F. Xiaolong, Y. fang, W. Haiyan, and L. Qixin, "Discussion on the construction of wisdom campus in universities under the backgroud of big data," *Journal of East China Normal University (Natural Science)*, vol. 3, 2015.
[4] W. Guoqiong, "Visualization of the analysis of big data on the behavior of college students," Master's thesis, Shandong University, 2016.
[5] L. Xuan, "Research and application of performance prediction model based on student behavior," Master's thesis, University of Electronic Science and Technology of China, 2017.
[6] H. Xiuquan, "A review of the research and application of campus card data," *Journal of Central China Normal University (Natural Sciences)*, no. S1, pp. 63–65, 2017.

[7] D. Haihui, "Research and implementation of campus card data mining based on hadoop," Master's thesis, Nanchang Hangkong University, 2017.

[8] Z. Dalong, "Design and implementation of campus card consumption analysis system based on hadoop," Master's thesis, Inner Mongolia Agricultural University, 2017.

[9] D. Rong, S. Xiaohui, and L. Zhiyong, "Analysis of consumer behavior of universities students with financial difficulties based on campus card," *Electronic Test*, no. 9X, pp. 78–79, 2016.

[10] C. Feng, "Ji yu xiao yuan yi ka tong xi tong de gao xiao yong hu jiu can xiao fei xing wei fen xi yu shu ju wa jue [analysis and data mining of dining consumption behavior of university users based on campus card system]," *The Chinese Journal of ICT in Education*, no. 5, pp. 47–49, 2014.

[11] J. Kaida, Z. Siyu, and S. Qiang, "Hadoop-based campus websites logging system design and implementation," *Journal of East China Normal University (Natural Sciences)*, vol. 2015, no. S1, pp. 126–131, 2015.

[12] P. Qi, "The research and implementation for college student behavior analysis system based on hadoop technology," Ph.D. dissertation, Beijing University of Posts and Telecommunications, 2015.

[13] Z. Chun, "Research on personalized book recommendation system based on hadoop platform," Master's thesis, Anhui University of Technology, 2017.

[14] Y. Honglei, "Design and implementation on visualization model for analysis of literature lending based on hadoop," Master's thesis, Changchun University of Technology, 2017.

[15] L. Ping, "Ji yu hadoop de k-means ju lei suan fa zai gao xiao tu shu guan gong zuo zhong de ying yong yan jiu [research on application of k-means clustering algorithm based on hadoop in the work of university libraries]," *Library & Information Science Tribune*, no. 32(05), pp. 35–41, 2014.

[16] W. Yongmei, "The construction of information resources in university libraries based on university users' network information behaviors," *The Journal of the Library Science in Jiangxi*, vol. 44, no. 3, pp. 29–32, 2014.

[17] L. Hai, "Research of management of historical data in college based on hybird storage," Master's thesis, Southeast University, 2015.

[18] M.-G. Jung, S.-A. Youn, J. Bae, and Y.-L. Choi, "A study on data input and output performance comparison of mongodb and postgresql in the big data environment," in *Database Theory and Application (DTA), 2015 8th International Conference on*. IEEE, 2015, pp. 14–17.

[19] Z. Haishan, "Dns query log analysis system based on open source software," *Journal of Xiamen University (Natural Science)*, vol. 2, p. 018, 2017.