MAPPING OF RELATIONAL AND OBJECT DATABASE TO NOSQL

Nguyen Dinh Thuan, Van Thi Phuong Lam
University of Information Technology, VNU-HCM
thuannd@uit.edu.vn, vanphuonglam@gmail.com

ABSTRACT: Today, NoSQL (Not Only SQL) has quite popular which solved the big data problem that the relational database is limited. In this paper, we propose the approach of the relational database and object model to NoSQL mapping. The main focus is on mapping object models to NoSQL. We used two methods for mapping: One is mapping one or many tables to NoSQL, one object or many objects to NoSQL. The other one is group objects are related to each other in the object-oriented data model, then map them to a collection of NoSQL. We successfully experimented with Mapping of Relational and Object Database model to NoSQL and we proved that using our approach can achieve a better performance.

Keywords: Object, Relation database, mapping, NoSQL.

I. INTRODUCTION

The Relational Database (RD) model has been proposed by E.F Codd since the early 1970s. This model has consistently as well as data integrity constraints. However, the amount of large and complex data has affected the performance of RD.

Object Relational Database Model (ORDM) [1] was proposed in the late 1990s. ORDBMS is reusable, reliable, capable of working with complex data types and enhancing performance over traditional data. ORDBMS limited in performance when dividing objects too complex to bring down the tables of relational database.

NoSQL (Not Only SQL) [2] is an open source and free. The feature of NoSQL is suitable for the project with big data, horizontally scaling, flexible schema, significant performance, high availability and fault tolerance. With the advantage of NoSQL, we propose methods for mapping of Relational and Object Database to NoSQL. The goal of the Mapping of Relational and Object Database to NoSQL is to improve performance with big data. In this paper, we have major contributions:

- Proposing a method of mapping a table or many tables of Relation Database to a collection or many collections in NoSQL. And mapping an object or many objects of the object-oriented model to a Collection or many Collections in NoSQL.
- Proposing a method of choosing and group objects are related to each other in the object-oriented data model, then mapping them to a collection of NoSQL that this collection has full information and relationships of Objects (based on the embedded Document model of NoSQL).
- Developing an application for Mapping of Relational and Object Database to NoSQL.
- Performance evaluation between traditional data model and NoSQL tool.
- The implementation includes .NET as the framework, C# language, MSSQL as the source relational database and MongoDB as the target database from NoSQL.

II. RELATED WORKS

Rejackets [3] proposed an abstract middleware approach on two databases to reduce the distance between SQL and NoSQL, bringing them closer together. Although users can understand and use the simple query system, but the performance isn’t high because of complicated nested data query.

Lee, Chao-Hsien, and Yu-Lin Zheng [4] propose an automatic mechanism for converting schemas from SQL to NoSQL by MySQL and HBase databases. Based on the test results, the author has proven that this proposal can improve access performance about 47%.

Tianfu Jia and his team [5] proposed a migration model approach and migration data from relational databases to MongoDB, from the proposed algorithm, demonstrating the NoSQL performance is better than the relational database performance.

Wolf, Florian, et al. [6] discussed and analyze the usage of a scalable NoSQL solution such as Basho’s RI AK as a backend for Hibernate. The authors pointed out that integrating RI AK into Hibernate is possible, but required several changes in the Hibernate core. The authors evaluated and compared the performance of RI AK and the performance of Hibernate-ORM [7] (configuration with MySQL) and provided a technique for integrating NoSQL into the ORM framework. The disadvantages of this paper are the measured times for bottom-up traversal do not reflect RI AKs potential, MapReduce support does not lead to additional speedup in our test scenarios.

U. Störl [8] gives an overview of the state-of-the-art in Java Object-NoSQL Mappers. U. Störl expect a contemporary ONM provides basic CRUD operations. However, its limitation is the supported query languages differ greatly in their expressiveness. During experiments, the author hasn't measured in cases unexpected, such as run into
cases when query operators had not yet been implemented, some query operators are not implemented with the semantics.

Another study of the Object-NoSQL Mappers by Reniers, Vincent [9] was also present a benchmark study quantifying and comparing the performance overhead for create, read, update and search operations. Reniers and Vincent measured performance of ONDMs and figure out that overhead is substantial for database operations in memory, but it isn’t substantial on-disk operations and high network latency. The ONDM API standardized mismatched with the technical capabilities of the NoSQL database. Performance overhead increases linearly with the number of results, Data Nucleus and Hibernate OGM’s search overhead are exceptionally high in comparison to the other ONDMs.

Bugiotti, Francesca, and Luca Cabibbo [10] proposed an ONDM(Object-NoSQL Datastore Mapper) framework. It provides application developers with a uniform programming interface (UPI) towards different systems. ONDM is based on the abstract data model and language to specify data representations (NoRM) [11], a system independent approach for NoSQL database design and the study consolidates knowledge on available ONDM frameworks [12]. This paper has been implemented a first prototype of ONDM but the architecture of this prototype is simple and it does not implement the NoAM language to specify custom data representations completely.

3. MAPPING OF RELATIONAL AND OBJECT DATABASE TO NOSQL

3.1. Database
We choose two Datasets: TPC-H and DBLP.

TPC-H includes 8 tables: Part, Supplier, Partsupp, Lineitem, Orders, Customer, Nation, Region.


3.2. Technology
We used .NET platform and C# language. Choose SQL Server to support for Relational Database, MongoDB to support for NoSQL databases. And drivers to support interactions with MongoDB using C# language. In addition, there is also support from the BSON library.

3.3. Proposed methodology
Method 1: Mapping a table or many tables of Relation Database to a Collection or many Collections in NoSQL. And mapping an object or many objects of the object-oriented model to a Collection or many Collections in NoSQL.

Method 2: Choosing and group objects are related to each other in the object-oriented data model, then mapping them to a collection of NoSQL that this collection has full information and relationships of Objects (Embedded Document model of NoSQL).

3.4. Data mapping process
3.4.1. Mapping of Relational to NoSQL
Dataset: choose any dataset from source of SQL Server (DBLP, TPC-H).

![Flow Diagram for Mapping of RD to NoSQL processing](Fig 1. Flow Diagram for Mapping of RD to NoSQL processing)

\[1\] http://www.tpc.org/tpch/
\[2\] http://dblp.uni-trier.de/
3.4.2. Mapping of Object Database to NoSQL

Dataset: TPC-H.

IV. EXPERIMENT

4.1. Experimental environment

For our initial setup, we installed both MongoDB 3.4.5 and Microsoft SQL Server 2014 Express on a machine running a Windows 10, 64bit Operating System, Intel Core i7-2630QM CPU 2.00GHz, RAM 6GB. Both databases were installed and stored their data on a SSD (Solid State Drive) for the fastest possible reads and writes. We wrote both test applications in C# using Visual Studio 2013. Visualization Tools: RoboMongo.

4.2. The application supports data mapping
Nguyen Dinh Thuan, Van Thi Phuong Lam

NoSQL model after mapping:

![Fig 5. Result of mapping of relation database DBLP to NoSQL](image)

![Fig 6. Result of mapping of Object TPC-H to NoSQL](image)

Based on result above, the mapping of relation and object to muscle has been successfully implemented. Specific tables, rows, columns, Primary keys, have been displayed in NoSQL.

V. PERFORMANCE EVALUATION BETWEEN SQL SERVER AND MONGODB

To evaluate the performance between SQL Server and MongoDB, an experiment was executed with INSERT, SELECT, UPDATE operations. Application Interface follows:

![Fig 7. Performance evaluation](image)

The following table demonstrates performance parameters for INSERT, SELECT, UPDATE.

<table>
<thead>
<tr>
<th>Number of Record</th>
<th>INSERT (millisecond)</th>
<th>SELECT (millisecond)</th>
<th>UPDATE (millisecond)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SQL Server</td>
<td>MongoDB</td>
<td>SQL Server</td>
</tr>
<tr>
<td>100</td>
<td>2.884</td>
<td>1.463</td>
<td>2.747</td>
</tr>
<tr>
<td>1000</td>
<td>27.508</td>
<td>14.628</td>
<td>27.990</td>
</tr>
<tr>
<td>10000</td>
<td>276.085</td>
<td>146.291</td>
<td>291.733</td>
</tr>
<tr>
<td>100000</td>
<td>2693.181</td>
<td>1442.994</td>
<td>3577.326</td>
</tr>
</tbody>
</table>
Chart of Performance on INSERT:

![Performance on INSERT](image1)

Fig 8. Performance Comparison for INSERT

Chart of Performance on SELECT:

![Performance on SELECT](image2)

Fig 9. Performance Comparison for SELECT Operation

Chart of Performance on UPDATE:

![Performance on UPDATE](image3)

Fig 10. Performance Comparison for UPDATE

The performance operations are done with different number of records ranging from 100 to 100,000 records. From these observations, we can see that MongoDB has better performance than SQL Server. Specifically, from 100 records to 10,000 records, MongoDB is twice as high as the Relation database (RD). From 100,000 records, MongoDB performed four times as high as RD. Given the parameters shown in the table above, MongoDB proved appropriate to work and process with large data.

VI. CONCLUSIONS AND FURTHER WORK

In this paper, the main task is to map relational and object relational databases to NoSQL, which focuses primarily on the object model mapping to NoSQL. We use two methods for mapping with C # .Net and driver...
interaction between MongoDB and C#. Through the performance parameters as well as model mapping, we have successfully implemented the data conversion process. The results have demonstrated that NoSQL performs better than relational and object-oriented data on the data set. This method can help businesses analyze and report business operations quickly and assist them in making effective business decisions.

NoSQL is suitable for large data, availability, high performance. However, the use of NoSQL does not have a common standard, but it depends on the actual situation that provides the best NoSQL implementation method.

REFERENCES