Migrating a web-application from a relational backend to a NoSQL data store.

AHMED ABDELKHALEK - K11720554, Julius Kepler University Linz
OMAR AMR - K11776960, Julius Kepler University Linz

With the current huge growth of data, Relational databases (RDB) are no longer capable of handling the demand. As, RDB are only capable of managing structured data and now-days data generated by most applications applications are becoming unstructured and harder to map and deal with in a relational way. That is why NoSQL are needed right now. But the issue what makes it harder to convert to the NoSQL is that the majority of websites had been already developed in SQL . Converting means re-writing the code that might cause huge amount of hassle and bugs which is time and money consuming.

Additional Key Words and Phrases: Relational Database, NoSQL, Big Data, Cloud computing

ACM Reference Format:
Ahmed Abdelkhalek - k11720554 and Omar Amr - k11776960. 2018. Migrating a web-application from a relational backend to a NoSQL data store.. 1, 4, Article 39 (February 2018), 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Applications today are dealing with huge amount of increasingly unstructured data e.g., the information collected from the Internet of Things (IoT). Previously, applications mainly depended on RDBMS, which can only handle structured data but not compatible for handling this kind of real-time data complexity. So that is why NoSQL databases are commonly used nowadays. It does not follow the same architecture as the RDBMS. NoSQL supports horizontal scaling, i.e., processing several instances on different servers simultaneously. A Basic classification of NoSQL (non-SQL) (i) Column, (ii) Document. (iii) key-value (iv) graph (v) multi-model. It is scalable and has high speed I/Os operations with lower cost Than the SQL. The issue here is how to actually migrate applications written based on RDBMS to support NoSQL databases without too much hassle. Section 2, talks about few possible techniques that can be used for migrating from SQL to NoSQL. Section 3 shows an example of how to try to do the actual migration using one of these techniques. Section 5 is a revision over the work done and concludes over the importance of the migration to NoSQL.

2 MIGRATION FROM SQL TO NOSQL

There are multiple techniques to migrate to NoSQL database. The idea to investigate which one suits your application the best and apply it. here you can find 7 techniques that we will be going over them.

2.1 Mid-Model [3]

This model is the one that was used in the example web-application in section 3. It is based on using the Data and query features. The first step is to extract the entity-relationship model from the source data. Then generate data features and query features of source data. After that, it generates mid-model based on extracted features and

Authors' addresses: Ahmed Abdelkhalek - k11720554, Julius Kepler University Linz, k11720554@students.jku.at; Omar Amr - k11776960, Julius Kepler University Linz, k11776960@students.jku.at.

XXXX-XXXX/2018/2-ART39 $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

source data model. Finally, it generates destination data model through pre-defined strategy and migrates data. Its main advantage is uniforming data for any NoSQL database but it needs information about the metadata.

2.2 NoSQLayer [5]

Which is an interface to automatically migrate from the rational database and the NoSQL database. This framework is divided into two modules, Data Migration Module which responsible for creating the equivalent structure from the original database to NoSQL data Model and Data Mapping Module which responsible for directing the queries from the DBMS to NoSQL. So it eliminated the need for any modifications in the application code while ensuring the efficiency from the qualitative and quantitative evaluations.

2.3 Content management systems (CMS) [2]

Which focus on denormalizing SQL database schemas and then migrate them into a NoSQL database autonomously. It is used in publishing websites (such system is like Word-press). The advantage of using this approach is flexibility while scaling up.

2.4 HBase database Technique [1]

it is a NoSQL database that runs over Hadoop. But one of its major limitations is that it requires a completely new design for the database when converting from an RDBMS to it. The migration process requires a lot of time and effort, however, it is very efficient for enormously large data.

2.5 Data adapter system [4]

It is a flexible system which has support for hybrid (Rational and NoSQL) databases architecture. It provides different modes for querying that user can choose from (BT, BD, BA). MapReduce approach is used. One drawback is that it may cause inconsistency in the data.

2.6 Automatic mapping framework [6]

Maintaining the features of the rational database but works only for MongoDB. The algorithms that the system uses requires access to the metadata. First stop is to create the MongoDB databases with inputs from the user. Then the framework creates these tables while checking the relations from the RDB with the specific steps specified in [6].

2.7 ZQL engine [7]

ZQL is an engine that is based on MySQL and hive. It aims to hide the specific details of both NoSQL database and RDBMS. One of its downsides that it may cause data replication.

3 PRACTICAL PART

3.1 Solution Preliminaries

3.1.1 PHP. PHP is an open source scripting language used mainly for web applications development. One of the main strengths of PHP is that it can be embedded inside HTML easily and the other way around. PHP is a fairly easy and simple language that contains a lot of powerful features. [http://php.net/manual/en/intro-whatis.php]

3.1.2 Apache. Apache is very popular and widely used web server software that provides HTTP services. It can be easily integrated with many common programming languages including PHP. [https://httpd.apache.org/]

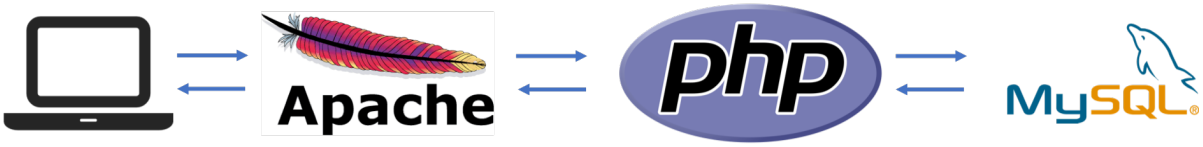

Fig. 1. Initial Integration Schema.

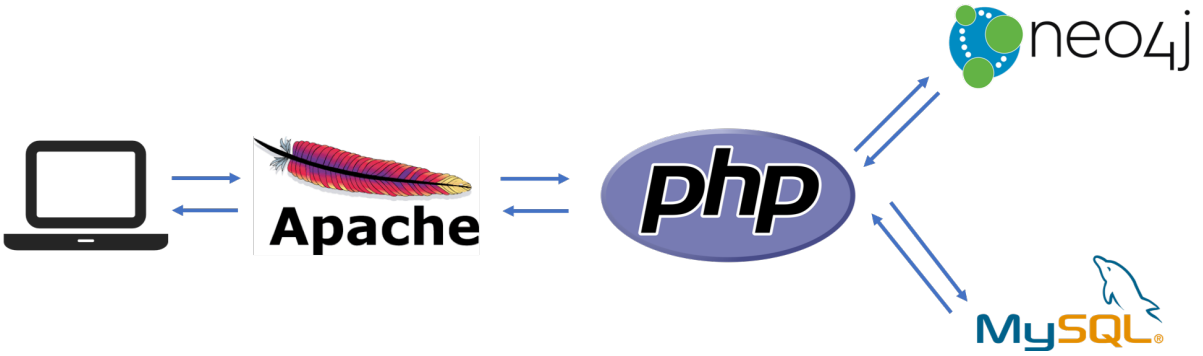

Fig. 2. Integration Schema after Connecting Neo4j.

3.1.3 MySQL. MySQL is a very popular and commonly used open source database management system. It is a relational database which stores data in separate tables in a structured way. It uses Structured Querying Language (SQL) to access and modify the database. [https://dev.mysql.com/doc/refman/5.7/en/what-is-mysql.html]

3.1.4 Neo4j. It is a NoSQL Graph Database that aims at decreasing the complexity of RDBMS when dealing with huge amounts of data. It consists of nodes which represents the entities and relationships that provide a direct connection between nodes in the graph database. https://neo4j.com/developer/graph-database/#_overview

3.1.5 Solution Implementation. Initially, a web application was created that uses RDBMS. PHP, Apache, and MySQL were used to create a very simple blog website that includes all the main operations in RDBMS such as insert, delete, and update. Figure 1 shows how technologies are integrated together to create the initial website. find a structured and organized way to integrate a NoSQL database. As illustrated previously, there are many models of doing that. Mid-Model was chosen to implement this integration as itâĂŹs a fairly simple model and there are no constraints on the type of NoSQL database that can be used with it. The first step was to define all entities and the relationships between those entities in order to be able to identify queries that extract all the features in the source MySQL database. Since the developed website is meant to illustrate the idea, it does not contain many features. The MySQL database contains two entities (Author, Article) and one many-to-many relationship (Writes) as an author can write multiple articles and an article can be written by many authors. The second step was to is to generate the mid-model which extracts the features from the source MySQL database and map them into the new Neo4j database. In order to do that, the Neo4j database needed to be integrated into the system first, which is a fairly easy process. Figure 2 shows the structure of the system after integrating the new database.

Fig. 3. MySQL Tables and Values.

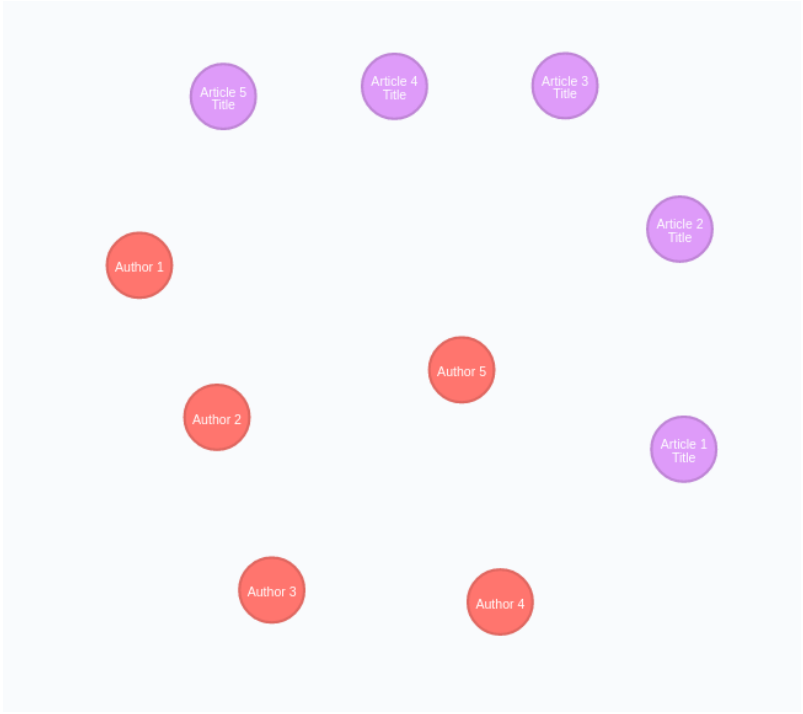

Fig. 4. Neo4j Database After Migrating Main Entities.

After achieving successful integration, a PHP script was used to extract data from MySQL and insert them into Neo4j according to the following steps:

(1) Extract tables that represent entities first from MySQL and then create a node for each entry in Neo4j containing all the information in this table. Entries of the same table have the same node type. For example, the two node types in the implementation are Author and Article.
(2) Get relations between entities in MySQL and create equivalent relations in Neo4j by connecting the two nodes with an edge.

Figures 3 shows the tables in the old MySQL database. While figures 4,5 shows how the data from MySQL database was mapped into Neo4j according to the pre-mentioned two steps.

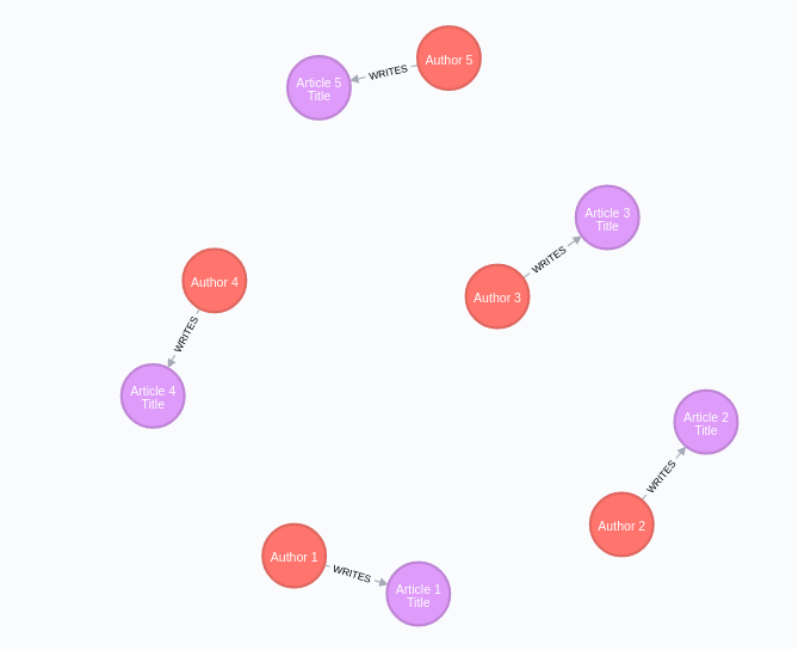

Fig. 5. Neo4j Database After Creating Relationships Between Main Entities.

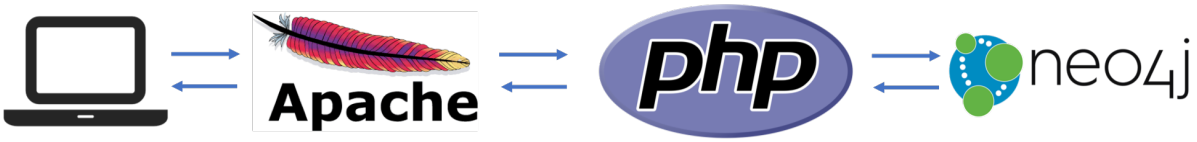

Fig. 6. Integration Schema After Removeing MySQL and Using Neo4j Only.

After all the MySQL database entries were transferred to Neo4j, MySQL database was removed from the system as shown in figure 6.

The final step was to modify the PHP code in order to replace the queries that used the MySQL database with other queries that use Neo4j database. This step is very simple and does not require any major changes in the code itself. The code follows the same logic and maintains the same code that is used for front-end display.

4 TESTING AND RESULTS

All the features that were included in the website before switching the database are tested to guarantee the success of the process.

4.1 Test1: Inserting new article to the database

The test successfully passed as shown in the highlighted part in figure 7.

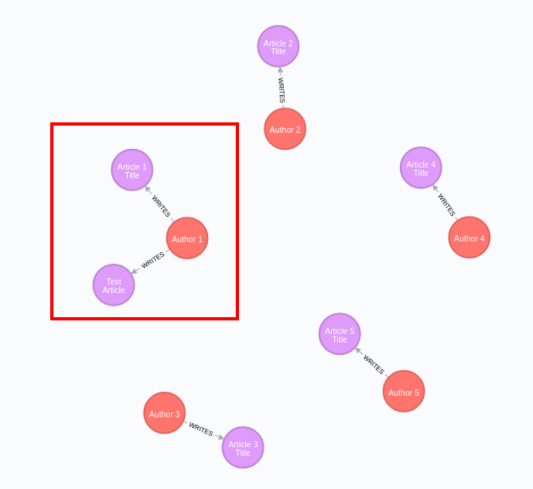

Fig. 7. Neo4j Databae After Insert Operation.

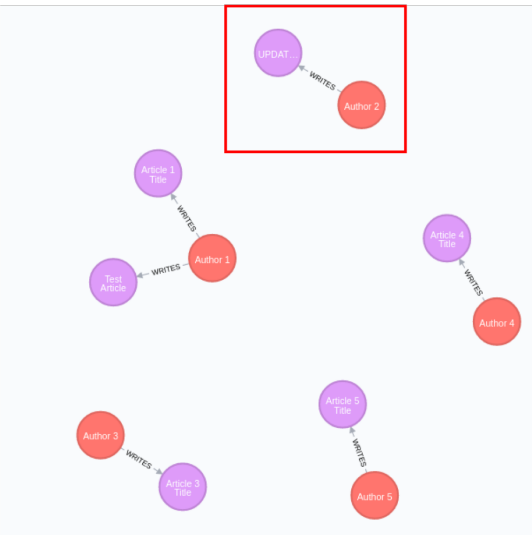

Fig. 8. Neo4j Databae After Update Operation..

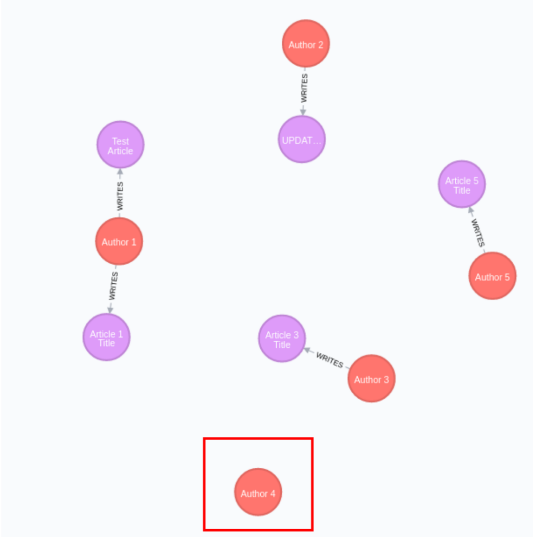

Fig. 9. Neo4j Databae After Delete Operation..

4.2 Test2: Updating an existing article

The test successfully passed as shown in the highlighted part in figure 8.

4.3 Test3: Deleting an article from the database

The test successfully passed as shown in the highlighted part in figure 9.

5 CONCLUSION

We conclude from what we showed in this paper that all techniques agree on how important is to use NoSQL in the current applications. As it can handle large volumes of structured, semi-structured and unstructured data. Not like RDBMS that can only handle structured data. NoSQL is also efficient, scalable and flexible architecture. It also has faster I/Os. And regarding the experiment, the migration process was not difficult neither time-consuming.

REFERENCES

[1] C. H. Lee and Y. L. Zheng. 2015. Automatic SQL-to-NoSQL schema transformation over the MySQL and HBase databases. In *2015 IEEE International Conference on Consumer Electronics - Taiwan*. 426–427. https://doi.org/10.1109/ICCE-TW.2015.7216979

[2] C. H. Lee and Y. L. Zheng. 2015. SQL-to-NoSQL Schema Denormalization and Migration: A Study on Content Management Systems. In *2015 IEEE International Conference on Systems, Man, and Cybernetics*. 2022–2026. https://doi.org/10.1109/SMC.2015.353

[3] D. Liang, Y. Lin, and G. Ding. 2015. Mid-model Design Used in Model Transition and Data Migration between Relational Databases and NoSQL Databases. In *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*. 866–869. https://doi.org/10.1109/SmartCity.2015.177

[4] Ying-Ti Liao, Jiazheng Zhou, Chia-Hung Lu, Shih-Chang Chen, Ching-Hsien Hsu, Wenguang Chen, Mon-Fong Jiang, and Yeh-Ching Chung. 2016. Data adapter for querying and transformation between SQL and NoSQL database. *Future Generation Computer Systems* 65 (2016), 111 – 121. https://doi.org/10.1016/j.future.2016.02.002 Special Issue on Big Data in the Cloud.

[5] Leonardo Rocha, Fernando Vale, Elder Cirilo, DÃąrlinton Barbosa, and Fernando MourÃčo. 2015. A Framework for Migrating Relational Datasets to NoSQL1. *Procedia Computer Science* 51 (2015), 2593 – 2602. https://doi.org/10.1016/j.procs.2015.05.367 International Conference On Computational Science, ICCS 2015.

[6] L. Stanescu, M. Brezovan, and D. D. Burdescu. 2016. Automatic mapping of MySQL databases to NoSQL MongoDB. In *2016 Federated Conference on Computer Science and Information Systems (FedCSIS)*. 837–840.

[7] J. Xu, M. Shi, C. Chen, Z. Zhang, J. Fu, and C. H. Liu. 2016. ZQL: A Unified Middleware Bridging Both Relational and NoSQL Databases. In *2016 IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing, 14th Intl Conf on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*. 730–737. https://doi.org/10.1109/DASC-PICom-DataCom-CyberSciTec.2016.129