Migration of Relational Databases to NoSQL - Methods of Analysis

Fábio Oliveira

Instituto Universitário de Lisboa (ISCTE-IUL), Lisboa, Portugal

Abílio Oliveira

Instituto Universitário de Lisboa (ISCTE-IUL), and Information Sciences,Technologies and Archictecture Research Centre (ISTAR-IUL), Lisboa, Portugal

Bráulio Alturas

Instituto Universitário de Lisboa (ISCTE-IUL), and Information Sciences,Technologies and Archictecture Research Centre (ISTAR-IUL), Lisboa, Portugal

Abstract

The amount of data to store, organize and manage in any organization, is very high and increases every day, fact well-known by companies as Facebook, Google or SAS. With this current growth rate, technologies must adapt to the amount of disposable data, and a new approach to information processing is required. Big Data technologies are more focused, and this is a reason for a greater spread of NoSQL database models. The purpose of this article is to validate the existing (and already used) migration methods and to adapt them, to understand the most efficient method to migrate a relational database to a NoSQL database. We will show the methodology used and what were the steps followed for the implementation, as well as the configuration of the environment used during the tests. Results show that in this migration process, the most efficient method is what is referred to as automatic offline migration. However, it requires a window of unavailability greater than the method of online migration, which in turn requires more resources from the operating system to migrate. Therefore, the most efficient method to migrate a database will depend on the application availability, and the computational resources available for it. We hope to make an important contribution in helping to choose a migration method to use, and the metrics that can be collected to better evaluate the performance of a migration.

Keywords: Migration, Methods, Metrics, Relational databases, NoSQL.

1. Introduction

Nowadays there seems to be data everywhere. Data that is organized and structured in information. How do I save, manage, transfer, and use them in a timely manner in a variety of contexts, particularly in large organizations or enterprises? It is estimated that the volume of data is growing 40% per year, and is expected to grow 44 times between 2009 and 2020 (Manyika et al., 2011).

Given the number of databases built based on the relational model, still predominant, it exemplifies and contextualizes the possibility of migration to the NoSQL (Not Only Structured Query Language) model, which are gradually adopted in medium and large companies.

We have the relational database and the NoSQL, both using methods that will impact the performance and the way we manage the database.

1.1 Relational database:

In 1985, Edgar Frank Codd published an article where he defined thirteen rules for a DBMS to be considered relational: (Codd, 1982, 1990)

1. Fundamental Rule
2. Rule of information
3. Access guarantee rule
4. Systematic treatment of null values
5. Online dynamic catalogue based on relational model
6. Comprehensive sub-language rule
7. View Update Rule
8. Inserting, updating, and deleting high level
9. Independence of physical data
10. Logical independence of data
11. Independence of integrity
12. Independence of distribution
13. Rule of non-subversion.

Also, every relational database must guarantee four characteristics in a transactional, that is known as ACID:

Atomicity: All the tasks of a transaction are executed or none of them are executed.

Consistency: The operation takes the database from one consistent state to another equally consistent.

Insulation: The effect of a transaction is not visible to other transactions until it is committed.

Durability: Changes made by committed transactions are permanent.

All the above features had a cost, this cost ends up generating a cost so that they are guaranteed, and this is necessary for the correct maintenance of the above materials.

1.2 NoSQL

The NoSQL are no relational databases and appeared as a solution of scalability to the relational model, because these do not follow the consistency ACID, but the CAP Theorem (Moniruzzaman & Hossain, 2013).

The CAP Theorem:

C – Consistency

A – Availability

P – Network Partition

BASE Property:

BA – Basically Available

S – Soft-State

E – Eventually Consistent

1.3 Migration of relational databases to NoSQL

Figure 1 lists some of the projects that have undergone migration from the relational model to the NoSQL model.

Organization	Migrated From	Application
eHarmony	Oracle & Postgres	Customer Data Management & Analytics
Shutterfly	Oracle	Web and Mobile Services
Cisco	Multiple RDBMS	Analytics, Social Networking
Craigslist	MySQL	Archive
Under Armour	Microsoft SQL Server	eCommerce
Foursquare	PostgreSQL	Social, Mobile Networking Platforms
MTV Networks	Multiple RDBMS	Centralized Content Management
Buzzfeed	MySQL	Real-Time Analytics
Verizon	Oracle	Single View, Employee Systems
The Weather Channel	Oracle & MySQL	Mobile Networking Platforms

Figure 1: Migrated databases from relational to NoSQL.
Source: (MongoDB, 2015)

In that article, we can see that there is no information about the migration methods, about the scenario configuration, the machine configuration or same the metrics collected during the execution, so we cannot have an investigation under this situation. We propose here a valid method that we can measure the migration a have same conclusion as performance, between other.

1.4 Migrated scenarios

Figure 2 lists the scenarios that we have used during our labs tests.

Figure 2: Migrated base Infrastructure:

- Scenario OLTP - On-line Transaction Processing - TPC-C.
  This scenario is compose basically by sort transaction and a complete reference can be obtain in the official documentation in tpc.org (TPC, 1992a).
- Scenario OLAP - On-line Analytical Processing - TPC-H.
  Here we have a data warehouse scenario, where there is aggregation and load process we also can check this one in the tpc.org (TPC, 1992b).
- Scenario HTAP - Hybrid Transactional Analytical Processing - TPC-C + TPC-H.
  This method is the previous two scenarios working together, a complete description is found in the Wikipedia (Wikipedia, 2017).

1.5 Migration methods

Figure 3 lists the migration methods indicates by the official developer of the NoSQL MongoDB, used as target during the migration.

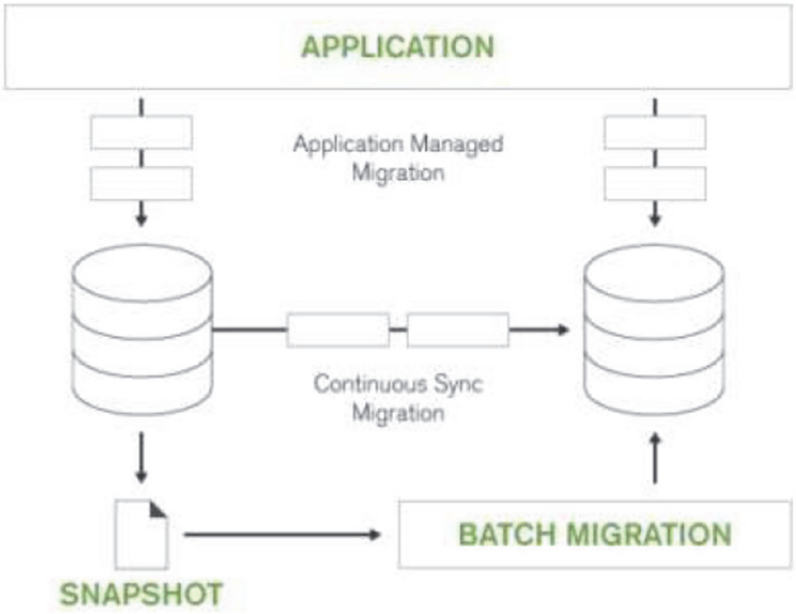

Figure 3: Migrated method by mongoDB:
Source: (MongoDB, 2015)

Our migration methods are based on the original documentation, we also include a new method, this one using online our skill to create a single migration code.

- Online migration with continuous synchronization.
  In this method, the application is always working and connected to the database, using the database resource in parallel with the migration.
- Offline migration through tool.
  Here we have the application completely down e all the computation power is used by the migration database and software.
- Manual offline migration through scripting.
  Also, we have the application completely down e all the computation power is used by the migration database and script executed by us.

1.6 Migration metrics

Table 1 lists the usage metrics in this work. We have collected a series of metrics that can provide us to answers a set of questions about the performance and the migration times.

Table 1: Migration metrics

Metric	*Main References*
Baseline	(Yaqub, 2012)
CPU	(Rodrigues, 2009; Yaqub, 2012)
Disk(I/O)	(Rodrigues, 2009; Yaqub, 2012)
Network	(Rodrigues, 2009; Yaqub, 2012)
Memory	added by the researcher
Latency	added by the researcher
Manual intervention	added by the researcher
Unavailability	added by the researcher

2. Main Objective

Emulate and analyze a database migration from relational database to NoSQL database, and with that result, we can answer what is the most efficient method for migrating a relational database to a NoSQL database?

3. Method

We have a case study of a qualitative nature, in which researchers define the starting point according to their own experience, or situations related to their practical life. Although in a case study different data collection techniques can be used, in this research we privilege the technique of observation, since we submit the data to tests and observe the results from it.

We carried out an extensive qualitative study with a sample of 9 cases of migration, where we started with three application scenarios to migrate using three different migration methods. In all 9 cases metrics were collected and the windows and migration were defined and which processes would be running in each window, so that with this division we could measure and analyze the proposed metrics.

4. Results

In summary, we have verified the performance of three possible migration methods between relational and NoSQL databases, using a set of metrics that provide us with a detailed view of resource consumption as well as details about the migration process.

From the theoretical point of view, the main contribution of this study is to show how methods of migrating relational databases to NoSQL and their associated metrics can be used, since these methods can be applied in several DBMS systems, not being linked only to this work, or to the software used here.

4.1 Identify the requirements and the various phases of a migration:

After the state of the art survey, several points were collected during a migration from relational database to NoSQL, based on other studies, as well as on processes that suppliers or software maintainers indicate as the way forward - being able to consult the reference in the listing below. The following points were gathered to be validated as requirements for a migration, coming from some research done, and others added by the author:

- Planning (Antaño, Castro, & Valencia, 2014; C. S. de Oliveira & Marcelino, 2012)
- Number of records / Initial situation (Antaño et al., 2014)
- Mapping the data types (Davenport & Dyche, 2013; Gomes, 2011; Pereira, 2014) Restrictions and triggering (Antaño et al., 2014)
- Character encoding (Antaño et al., 2014; Neto, Neto, Junior, & Oliveira, 2013)
- Tests (added by the researcher)
- Implementation (added by the researcher)
- Partial and Final Monitoring / Validation (added by the researcher)
- Staging area / No staging area (added by the researcher)
- Failures during migration (added by the researcher)
- Data modelling (Gomes, 2011)

We can divide the migration method in phases, as there is four phases for the online method and two for the offline methods.

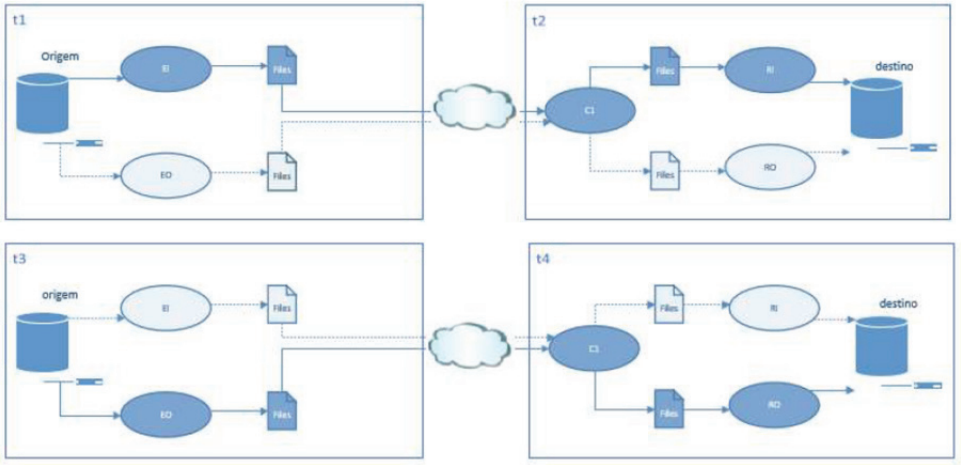

Figure 4: Phases of the online migration with continuous synchronization method:

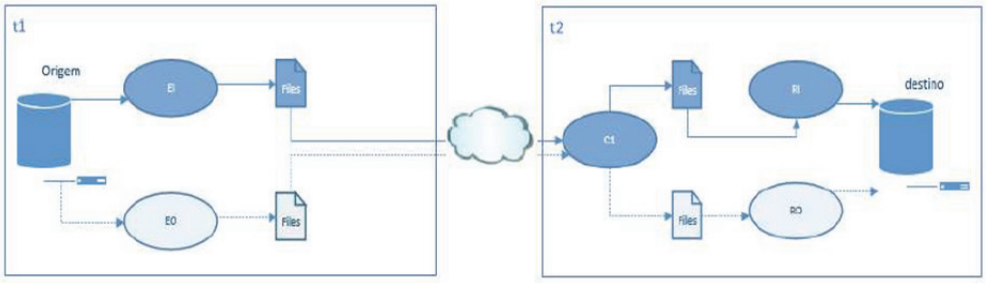

Figure 5: Phases of the offline migration through tool method:

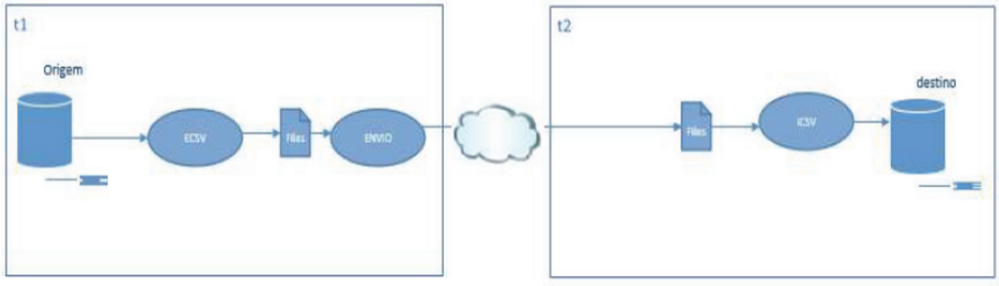

Figure 6: Phases of the manual offline migration through scripting method:

4.2 Check the possible methods of migrating from a relational database to a NoSQL.

To carry out a literature review, to research in theses of the area, to validate the documentation of the main software developers of the thematic area and to filter those that best assist in the framing of this work. With that done, we worked with three migration methods as describe in the chapter "1.5 *Migration methods*".

4.3 Evaluate each of these methods according to the steps followed in each phase of the migration - using specific and appropriate metrics in each case.

In order to respond to this objective, we analyzed the metrics collected in each of the phases, a small sample of these metrics will be visualized here, and however, all the metrics can be analyzed in the work done in (F. V. de Oliveira, 2017).

In figure 7 we can see the CPU consumption during the first phase of the OLTP scenario migration through the online migration method.

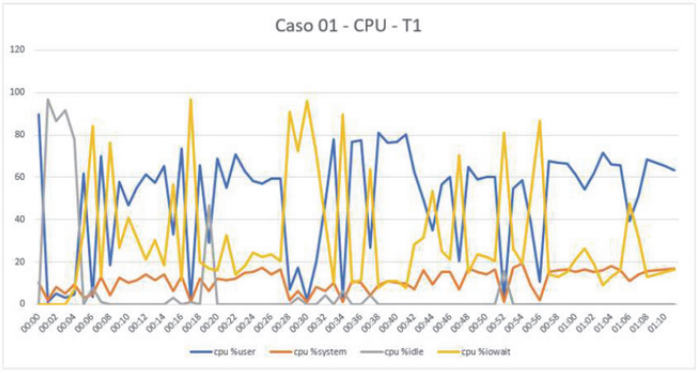

Figura 28 - Caso 01 - CPU - T1

In this metric, we can clearly see the amount of CPU consumed by the application, which is identified by the "CPU% user" event, and we can also see the amount of CPU consumed by the operating system, the amount of CPU responsible for the I / O used.

In figure 8 we have the disk activity, that is, how much writing activity and how much reading activity we have in this phase of the migration, and here we can notice the reading behavior.

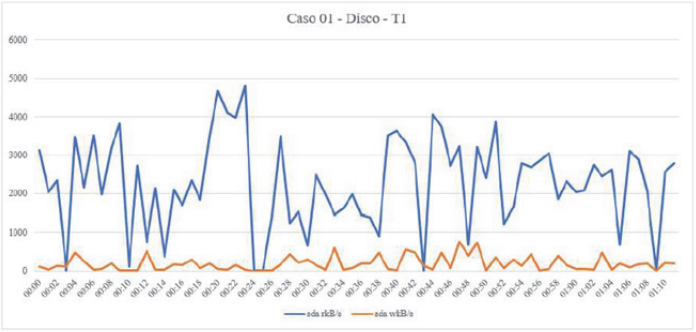

Figura 31 - Caso 01 - Disco - T1

4.4 Compare the various methods according to the established metrics.

We can compare each phase against any other method in the same phase, with that, we can evaluate which phase consume more resource in any phase, of course, comparing the same phase in the methods here provided.

4.5 Determine the most efficient migration method for each of the studied database scenarios.

The method of offline migration through tool has always proved to be the fastest migration method.

We can thus answer that the most efficient migration method is the automatic offline migration

method. However, this requires a large window of unavailability, and if it is not possible to stop the application or even the database during the migration process, the online method will be the one indicated. To answer this question, we must first ask the following question:

"What window of unavailability is available for the migration?"

5. Conclusions

We have verified the performance of three possible migration methods between relational and NoSQL databases, using a set of metrics that provide us with a detailed view of resource consumption as well as details about the migration process.

From the theoretical point of view, the major contribution of this study is to show how methods of migrating relational databases to NoSQL, and their associated metrics can be used, since these methods can be applied in several systems.

We can thus respond, after applying and analysing metrics in migration cases and migration methods proposed here, that the most efficient migration method is the automatic offline migration method. However, this requires a large window of unavailability, and if it is not possible to stop the application or even the database during the migration process, the online method will be the one indicated. With this conclusion, we understand that it is always necessary to validate resources and unavailability as requirements of the project and not the migration itself, because it can happen without or with unavailability, however, with direct reflection in migration times.

References

Antaño, A. C. M., Castro, J. M. M., & Valencia, R. E. C. (2014). Migracion de Bases de Datos SQL a NoSQL. *Revista Tlamati*, *Especial 3*, 144–148.

Codd, E. F. (1982). Relational database: a practical foundation for productivity. *Communications of the ACM*, *25*(2), 109–117.

Codd, E. F. (1990). *The relational model for database management: version 2*. United States of America: Addison-Wesley.

Davenport, T. H., & Dyche, J. (2013). *Big Data in Big Companies*. *International Institute for Analytics*.

Gomes, P. F. L. (2011). *Migração de aplicações legadas para bases de dados NoSQL*. Universidade do Minho.

Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., & Byers, A. H. (2011). Big data: The next frontier for innovation, competition, and productivity. Retrieved January 9, 2016, from https://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/big-data-the-next-frontier-for-innovation

MongoDB. (2015). RDBMS to MongoDB Migration Guide. *A MongoDB White Paper*, 16.

Moniruzzaman, A. B. M., & Hossain, S. A. (2013). NoSQL Database : New Era of Databases for Big data Analytics-Classification , Characteristics and Comparison. *International Journal of Database Theory and Application*, *6*(4), 14.

Neto, P. de A. dos santos, Neto, J. R., Junior, F. das C. R., & Oliveira, P. A. (2013). Requisitos para ferramentas de Migração de Dados. In *IX Simpósio Brasileiro de Sistemas de Informação* (pp. 887–898). Teresina.

Oliveira, C. S. de, & Marcelino, M. A. (2012). Metodologias e Extratégias de Migração de Dados. *Sinergia (CEFETSP)*, *13*(3), 183–191. Retrieved from www2.ifsp.edu.br/edu/prp/sinergia

Oliveira, F. V. de. (2017). *Migração de bases de dados relacionais para NoSQL - Métodos de Análise*. ISCTE-IUL Instituto Universitário de Lisboa.

Pereira, D. J. P. (2014). *Armazens de dados em bases de dados NoSQL*. Instituto Superior de Engenharia do Porto.

Rodrigues, R. A. B. (2009). *Métricas e Ferramentas Livres para Análise de Capacidade em Servidores Linux*. Universidade Federal de Lavras.

TPC. (1992a). TPC-C is an On-Line Transaction Processing Benchmark. Retrieved May 29, 2017, from http://www.tpc.org/tpcc/

TPC. (1992b). TPC-H is a Decision Support Benchmark. Retrieved January 11, 2016, from http://www.tpc.org/tpch/

Wikipedia. (2017). Hybrid Transactional/Analytical Processing (HTAP). Retrieved June 22, 2017, from https://en.wikipedia.org/wiki/Hybrid_Transactional/Analytical_Processing_(HTAP)

Yaqub, N. (2012). *Comparison of Virtualization Performance: VMWare and KVM*. *Signal Processing*. Univesity of Oslo.