

People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

KASDI Merbah University - Ouargla

Faculty of New Information and Communication Technologies



Professional Master's Degree

Domain: Computer and Information Technology

Specialty: Industrial computer science

Presented by: Tihami Sadia

Belgmari Nacira

Theme:

Switching from relational databases to NoSql databases

on 22/06/ 2021 in front of the jury composed of :

.Ben Kherourou Chafika

. Koraichi Wasilla

. Mahdjoub Mohmed Bachir

Supervised

Examiner

Examiner

University KM Ouargla

University KM Ouargla

University KM Ouargla

School year: 2020/2021

Recognition

We would first like to thank God who gave us all these years, health, courage and the will to do this work. Big thanks to you for pointing us on the right path for helping us throughout our school years and for all the grace that surrounds us.

We warmly thank our supervisor, Ms. Ben Kherourou Shafika Siham, for accepting our supervision and for all her assistance and support during the investigation. I express my deep gratitude to her for her patience, valuable advice and invaluable support. A big thank you to our families, especially our parents who have supported and followed us throughout this project. To our dear friends who have always been present and loyal.

Finally, we would like to thank the discussion panel members who did a good job.

Dedication

With great love and honest words.

*I thank God who helped me in writing and completing this
memorandum.*

*To the greatest person in my life and my role model, my beloved
mother and the dearest of people, my beloved father, to my sisters
Fatima, Djamilia, Khaira, Razika and Enas and my brothers
Ahmed and Tayeb, and to my sister's children, My brother's wife,
and my sister's husband, my niece and my friends Nawal,
Mubaraka, Akila, Zainab and Samah, Habiba, Asmaa and
Rima, and special thanks to Sadia Tihami, who shared the work
with me, and to my classmates, and special thanks to Djeriou
Abdel Hakim, who supported me. I thank everyone who helped me,
with my greetings to all.*

Nacira Belgmari

Dedication

With great love and sincere words.

I thank God who helped me write and complete this note.

Then I would like to express my thanks to the greatest person in my life and role model, my beloved mother and dearest people, my beloved father, and to my sisters Samira, Zahia, Yasmina, Fatima, Madiha, Dalal Kawthar and the children of my sisters and brothers Ismail, Ali, Abd El Razak, Abd El Satar, Muhammad El-Azhar and their wives and children Each in his name and my friends each in her own name, and a special thank you to Nacira Belqmari, who shared the work with me, in addition to a special thanks to my sister, Yasmina, for her material and moral support, and to my classmates. I thank everyone who helped me even in the simplest things, with greetings to all

Sadia Tihami.

Abstract:

Over the past decades the relational database has been responsible for managing data in applications, Relational databases are rapidly approaching their limits , Following this issue, new technologies such as NoSQL databases have arisen, which fundamentally alter the structure of databases we are accustomed to seeing, allowing for greater performance and availability of services. this paper aims to find the best system for managing the data that is pumped In large quantities, and since the NoSQL database is one of the most important bases that can manage big data. A method for migrating a relational database to a column-oriented NoSQL database is provided (DBNOC). The translation of the source schema into the target schema and the transformation of the data are the two key steps of this technique. In terms of data transformation, it includes data extraction, formatting, and injection from a source database.

Keywords : managing data; NoSql; Relational Database; DBNOC; MongoDB .

Résumé :

Au cours des dernières décennies la base de données relationnelle a été chargée de gérer les données dans les applications, Les bases de données relationnelles approchent rapidement de leurs limites, Suite à ce problème, de nouvelles technologies telles que les bases de données NoSQL sont apparues, qui modifient fondamentalement la structure des bases de données que nous sommes habitués à voir, permettant une plus grande performance et disponibilité des services. cet article vise à trouver le meilleur système pour gérer les données qui sont pompées en grande quantité, et puisque la base de données NoSQL est l'une des bases les plus importantes pouvant gérer le big data. L'invention concerne un procédé de migration d'une base de données relationnelle vers une base de données NoSQL orientée colonnes (DBNOC). La traduction du schém.

source dans le schéma cible et la transformation des données sont les deux étapes clés de cette technique. En termes de transformation de données, il comprend l'extraction, le formatage et l'injection de données à partir d'une base de données source.

Mots-clés : gestion des données ; NoSql ; Base de données relationnelle ; DBNOC ; MongoDB.

الملخص:

على مدى العقود الماضية ، كانت قاعدة البيانات العلائقية مسؤولة عن إدارة البيانات في التطبيقات ، وقواعد البيانات العلائقية تقترب بسرعة من حدودها ، وبعد هذه المشكلة ، ظهرت تقنيات جديدة مثل قواعد بيانات NoSQL ، والتي تغير بشكل أساسي بنية قواعد البيانات التي اعتدنا على رؤيتها ، السماح بأداء وتوافر أفضل للخدمات. تهدف هذه الورقة إلى إيجاد أفضل نظام لإدارة البيانات التي يتم ضخها بكميات كبيرة ، وبما أن قاعدة بيانات NoSQL تعد من أهم القواعد التي يمكنها إدارة البيانات الضخمة. يتم توفير طريقة لترحيل قاعدة بيانات علائقية إلى قاعدة بيانات NoSQL الموجهة نحو الأعمدة (DBNOC). تعتبر ترجمة مخطط المصدر إلى مخطط الهدف وتحويل البيانات الخطوتين الأساسيتين لهذه التقنية. فيما يتعلق بتحويل البيانات ، فإنه يشمل استخراج البيانات وتنسيقها وحقنها من قاعدة بيانات المصدر.

الكلمات المفتاحية: إدارة البيانات, لا قاعدة بيانات علائقية MongoDB. DBNOC.

Summary

Abstract

General introduction:	1
Chapter 1:	3
Relational database and NoSQL	3
1.1.Introduction:	4
1.2. DataBase:.....	4
1.2.1 Definition of the database :.....	4
1.2.2 Database types:	4
1.2.3 Functions of Database :	5
1.2.4 Data Structure Concept:	6
1.2.5 Objectives of database management systems :	7
1.3.Data :	7
1.3.1 Definition of Data :	7
1.3.2 Definition Big Data :.....	8
.14. Relational Databases :	8
1.4.1 Limitations of the relational model :.....	8
.14.1 Definition of Relational databases :	8
.14.2 Advantages :	8
.14.3 Disadvantages :.....	9
.14.4 Relational database properties :.....	9
.15. NoSQLDataBases :.....	10
.15.1 Definition of NoSQLdatabases :.....	10
.15.2 L'´emergence du NoSQL :.....	10
.15.3 Features of the NoSQL model:	10
.15.4 Advantages :	11
.15.5 Types of NOSQL databases:	12
.15.6NoSQL Database Properties :	14
.16. A comparison between a relational database and a NoSQLdatabase :	14
.17.Some current Solutions:	16
.18.The Tools Used For Migration :.....	16
1.8.1 Hadoop :	17

.18.2.MapReduce :	20
.18.3 Hbase :	21
.18.4 MongoDB :	22
.18.5 Cassandra :	22
1.9. Conclusion:	24
Chapter 2:	25
Approach for passage	25
2.2. Source Data Model:	26
2.3. Objective Data Model:	28
2.3.1 DBNOC Data Structures: Table	29
2.3.2 DBNOC Data Structures: The Key	29
2.3.3 Data Structures in DBNOC: The Column Family	29
2.3.4 DBNOC Data Structures: Simple Type Columns	30
2.3.5 DBNOC Data Structures: Supertype Columns	31
2.4 Translating the source data model into a target model:	32
2.4.1 Entity translation:	32
2.4.2 Translating multiple to one association:	33
2.4.3 Translating a many-to-many association:	34
2.5 Convert data from RDB to DBNOC:	35
2.6.Key correspondence table:	36
2.7. Conclusion:	36
Chapter3 :	37
Implementation	37
3.1. Introduction :	38
3.2. Definition of the MongoDB :	38
3.3. Definition of the Mysql :	38
3.4.Installing single node MongoDB:	38
3.4.1 Getting ready:	38
3.5. Starting the MongoDB Client:	40
3.6. Our work steps :	42
3.7. Migration steps:	43
General Conclusion :	46
Library :	47

List of Figures

Figure I.1 : Document-oriented databases.....	12
Figure I.2 :Graph-oriented databases	13
FigureI.3 :Column-oriented databases.	13
Figure I.4 : Schematic of a database oriented document.....	18
FigureI.5 : Schematic of a column-oriented database.	19
Figure I.6 :Schematic of a graph-oriented database.....	19
Figure I.7 : HDFS Schème.	20
Figure I.8 : Data replication using HDFS.....	20
FigureI.9 : Functioning of MapReduce.	22
Figure II.1: Relation many to many.....	28
Figure II.2 : Relation many to many.....	28
Figure II.3 : Relation one to one.....	29
Figure II.4: Table data structure in DBNOC.....	30
Figure II.5 :Modeling a family of columns.....	31
Figure II.6: Sample column of simple type in DBNOC.....	31
Figure II.7: Example of storage representation according to the DBNOC model.....	32
Figure II.8 :Supercolumn modelling in DBNOC logic.....	32
Figure II.9 : Example of a relationship one to several.....	35
Figure II.10: Translation of the many-to-many relationship.....	36
Figure II.11 : Conversion from RDB to DBNOC.....	36
Figure II.12: The main correspondence table.....	37
Figure III.1 Dowling of MongoDB.....	40
Figure III.2 : Dowling of MongoDB.....	40
Figure III.3 :the directory of MongoDB	41
Figure III.4 :the Run of Mongod	41
Figure III.5 :the Run of Mongo	42
Figure III.6 :Activationmongodb.....	42
Figure III.7:ActivationMongoDBCompass.....	43
Figure III.8:Log in to Mysql.....	43
Figure III.9:insert the data into each table.....	44
Figure III.10:Create a database in MongoCompass.....	45
Figure III.11:database looks like after it was created.....	45
Figure III.12:Adding the ADD DATA to the CSV file.	46
Figure III.13:After downloading the CSV file.....	46
Figure III.14: the database after the migration to NoSQL.....	47

Liste des des tableaux

table I.1 : Comparison table between relational database and NOSQL database:	14
table II.1 : DBNOC Entity Model	33
table II.2 : Example data representation of a many-to-one association.....	34
table II.3 : Data model for a many-to-one association	34

General introduction:

Many organizations and businesses are dealing with a database explosion. These blocks of data, which are routinely processed and stored, degrade performance, prompting different companies to seek solutions by expanding their infrastructures. Databases are one of the most crucial services for all departments and corporations, as they serve as the company's collective memory, storing all of the company's critical information.

Unfortunately, the problem stems mostly from the model used to describe the data. . The majority of published databases are relational in design, which doesn't exactly suit with the evolution and expansion capabilities of this sort of distributed environment. Big data refers to a set of approaches for increasing the amount, variety, and volume of data. Big Data concerns are part of a complicated setting that sits at the intersection of two major concerns:

Implementing new mass storage technologies; capturing data at fast speeds and in real time, if possible;

To address these three issues, new storage options are available and various data management solutions have emerged. By proposing alternatives to the relational paradigm, the so-called "NoSQL" takes its place. In large-scale systems, NoSQL databases provide novel storage options. Many users of traditional SQL DBMS desire to convert to these new NoSQL alternatives in order to prepare for future data explosions. But don't want to start with blank databases. Instead, learn how to access data from relational databases.

We propose a way to convert a relational database to a NoSQL MongoDB database as part of our PFE. The suggested technique will allow us to:

-Migrating relational databases to NoSQL databases.

-Handle and host the NoSQL system.

The following is how our thesis will be structured:

Following an introduction, Chapter 1 will provide an overview of relational databases, as well as the constraints that these models have reached in distributed contexts and for NoSQL technology, as well as the many types of NoSQL databases and their benefits. We try to defend our choice of MongoDB as a NoSQL platform. The latter will be discussed in greater depth further down.

The second Chapter will cover RDBMS migration and implementation. As a result, the focus of this work will be on the design of the RDBMS-NoSQL migration model, specifically the migration approach from the relational data model to the column-oriented data model, that is, to convey a systematic process view, a framework that introduces the various stages of the transformation from the conceptual data model (CDM) to Column oriented datagram.

In the third Chapter, we will present the migration from Relational Database to DBNOC in MangoDB.

Finally, we give a brief conclusion of our work and offer an outlook on future researches.

Chapter 1:

Relational database and NoSQL

1.1.Introduction:

In this first chapter, we covered the following elements: the database, the concept of structure, the definition of data, and big data, in addition to the definition of relational databases, and NoSQL databases, and the characteristics, advantages and disadvantages of each of them separately, and in the goodness of a comparison between databases Relational Data and NoSQL Database.

1.2. DataBase:

1.2.1 Definition of the database :

It is a group of related digital information that is organized in a table or a group of tables, each of which may contain a field or several fields. For example, we use logging into a website on the Internet. The site uses the database to store personal data such as name, e-mail, phone number, etc. The database is organized for the possibility of storage, deletion and modification easily. Most of the time the database is managed by a database management system (DBMS).[1]

1.2.2 Database types:

We can classify database into four kinds which are [2]

- a. **Bibliographic Databases:** having data that is unformatted (unformatted data). They're made up of textual material that, by definition, has little or no formatting. In libraries and information systems, such databases are frequently utilized. Here, data could be made up of book abstracts and other publications with key terms and phrases. The abstract can be used to judge whether or not the document is of interest.
- b. **Knowledge Databases :** Artificial Intelligence (AI) applications employ them. The information in these files is distinct and formatted. There are generally many different types of data in these, with just a few instances of each type. The size of the data in such databases is as vast as the definition of the data.
- c. **Graphic-Oriented Databases :** could be useful in computer-assisted design (CAD). The data in such a database is referred to as "active." This signifies that data is a technique that can be carried out. Because the aforementioned **aandb** cannot be carried out by a machine, any changes to data can be made.

Ex : Computer-Aided Design (CAD)

Computer-Aided Learning (CAL)

Computer-Aided Instruction (CAI)

- d. Decision-making Databases :** are employed in corporate management and related administrative functions. One may use the data in these databases to solve problems like resource planning and sales forecasting. These databases are distinguished by the fact that they contain the following data:
- i. Formatted
 - ii. Far longer than description
 - iii. Passive

Decision-making databases are frequently referred to simply as databases. Database Management Systems (DBMS) are classified according to the type of databases they manage, such as bibliographic database management systems, knowledge database management systems, and so on.

1.2.3 Functions of Database :

The overall goal of a database is to manage data as a whole. The overall goal is to make information access simple, quick, affordable, and adaptable for users.[2]

- **Controlled redundancy :** The data uses excess space and is therefore ineffective. System performance is increased by controlling redundancy.
- **User-friendly :** The ease with which a user-friendly database package can be learned and used is a key aspect.
- **Data independence :** It allows changes to be made at one level of the database without affecting the others, for example, changing hardware and storage technologies or adding new data without rewriting the application program.
- **Economy :** It is important to use, store and modify data cheaply.
- **Accuracy and integrity :** Even if the duplication is removed, the database may still contain inaccurate information. Avoiding these situations is easier with central database control. The consistency of data and content quality is ensured by the accuracy of the database. Data inaccuracies are disclosed by safety controls when they occur.
- **Recovery from failure :** When a database has multi-user access, the system must recover promptly after a failure so that no transactions are lost. It aids with the accuracy and integrity of data.

Chapter1

- Privacy and Security** : Security measures must be implemented in order for data to remain private. DBMS ensures sufficient security by centralized control to prevent unauthorized access, i.e. complete sovereignty over operational data.
- Performance** : It underlines that the length of time it takes to respond to enquiries that are appropriate for data use is determined by the nature of the user-database dialogue.
- Database retrieval, analysis, storage** : It makes database retrieval, analysis, and storage much easier.
- Compatibility** :Usefulness refers to the ability of hardware and software to work with a variety of computers.
- Concurrency control** : is a feature that lets several users to access a database at the same time while maintaining data integrity.
- Support** : Complex file structure and accessibility are supported. Mark, for example.
- Data Sharing** : The database allows any number of users to share data under its control.
- Standards can be enforced** : Standardizing data formats in storage is especially useful for data exchange between systems.

1.2.4 Data Structure Concept: [2]

Data is organized according to the Data Model. A collection of data pieces that are treated as a single entity. Book details, for example, is a data structure made up of the following data elements: Author name, Title, Publisher name, ISBN, and Quantity. Although all DBMSs have a similar approach to data management, they differ in one important aspect: data structure.

Analyzing the logical structure of data in complicated databases can be done in a variety of ways.

There are three types of data structure, viz

1.2.4.1 List Structure : A list is nothing more than a specific data structure made up of data records in which the Nth record is associated to the previous two records (N-1) and (N-2). This establishes a one-on-one relationship.

1.2.4.2 Tree / Hierarchical Structure :A tree structure is a non-linear multilevel hierarchical structure in which each node has N-nodes at any level below it. However, there is just one node in the structure above it.

The entrance is from the top, the search or passage orientation is downhill, and there are no branches on the tree trunk (touch).

Chapter1

1.2.4.3 Network Structure: Another type of hierarchical structure is network structure. The data is represented by records and connections in this view, just as it is in the hierarchy approach. A network, on the other hand, is a broader structure than a hierarchy. Relationships between entities are possible with a network structure .The database is viewed as a collection of discrete record occurrences, with each node having any number of subordinate nodes. A graph structure is the same as a network structure. Many-to-many relationships are formed as a result of this. Sets are the relationships between the various items.

1.2.5 Objectives of database management systems :[1]

1.2.5.1 Ease of data management :

DBMS is a system that provides structures for storing, managing, manipulating and sharing data, as well as organizing external presentations, and DBMS allows modification, deletion or any other operation with ease. The system is also characterized by confidentiality and honesty.

1.2.5.2 Data Sharing :

The goal here is to find a method that allows applications to share database data when they want to access it, but at some point The data seem to be the only ones to use them without even being, but also without knowing it .Another application can modify them at the same time.

1.2.5.3 Data security :

This objective has two parts. First of all, the data must be protected against unauthorized or malicious access. Appropriate mechanisms must be in place to authorize, control or deactivate the access rights of any user to any set of data.

. On the other hand, data security must also be guaranteed in the event of a program or system failure, or even hardware and machine failure.

1.3.Data :

1.3.1 Definition of Data :

To put it another way, data can be facts about any object under investigation. Your name, age, height, weight, and other personal information are examples of data about you. Data can also be a picture, an image, a file, a pdf , and so on.

1.3.2 Definition Big Data :

The concept of processing large information in terms of large volume, high velocity and large variety, to extract value. These data require efficient and innovative forms of information processing to enable a better decision making .in order to improve decision-making and process control. [3]

.14. Relational Databases :

1.4.1 Limitations of the relational model :

Databases have been established for over 50 years, and the relational model has been around for around 40 years. This architecture, although incredibly strong, has limitations that certain services/sites, such as Google, Facebook, and others, have long reached. Indeed, because this type of site contains tens of millions or even billions of items in its databases, as well as tens of thousands of daily visitors, a single system cannot handle them all.[4]

This approach has many limitations, including:

- The problem of managing heterogeneous things
- Problem related to CAP Theory
- Non-optimal query problem due to the use of joins
- Problem related to the application of acid properties in a distributed environment
- Problem splitting vertical data

.14.1 Definition of Relational databases :

Relational databases appeared in 1970. It is one of the types of databases that organizes data points related to each other for easy access. Relational databases rely on relational models. Data is structured in tables. Each row in the table is a record. In addition to table columns containing Data attributes, relational databases also use structured query language (SQL).[5]

.14.2 Advantages :

Methodology for rigorous design (normalization, set theory)

- All other database structures can be boiled down to a collection of relational tables.
 - Network and hierarchical approaches are used to store and retrieve data in mainframe databases.
 - Data access is controlled by a set of hard-coded rules.
 - Without a pre-defined access path, extracting data from this type of

Chapter1

database is quite challenging.

- Fast retrieval times for multi-user transactions

Environment.

- Easier to use than other database systems
- Editable - new rows and tables can be added simply
- The procedure for joining relational data
 - A group is a group of elements in common with each member having some unique traits or features, according to the theory of algebraic groups.

- Very adaptable and strong

- The processing time is short

- Processors that are faster, multi-threaded operating systems, and parallel servers

- Indexes, fast networks, and disk arrays in clusters
- 57,000 users (Oracle/IBM) at the same time

.14.3 Disadvantages :

- Expensive solutions that necessitate careful consideration.
- If sufficient data analysis is not performed prior to implementation, it is easy to develop poorly planned and inefficient database designs.

.14.4 Relational database properties :

Relational databases have the following properties :[6]

-Values are atomic.

-All of the values in a column have the same data type.

-Each row is unique.

-The sequence of columns is insignificant.

-The sequence of rows is insignificant.

-Each column has a unique name.

-Integrity constraints maintain data consistency across multiple tables.

.15. NoSQL DataBases :

.15.1 Definition of NoSQL databases :

NoSQL databases allow redundancy to better meet inflexible needs, fault tolerance, and scalability, and are suitable for ultra-fast and ultra-fast processing storage, such as Like: MongoDB, Cassandra, or Redis. It is considered the latter, which implements more efficient storage systems than traditional SQL for analyzing group data (key / value prompt, document, column or graph database). [7]

.15.2 L'`emergence du NoSQL :

As the internet grows, businesses are deploying MySQL teams dedicated to pushing the boundaries of relationship models. But most businesses cannot afford to use these resources. For the latter, the relational model is no longer the guarantee of ease of use, and it is now time to change NoSQL structures and formats. In addition, the use of parallel and distributed systems is becoming increasingly popular. [1]

.15.3 Features of the NoSQL model: [1]

•The CAP Theorem or Brewer's Theorem:

The NoSQL model is based on a blend of three characteristics: consistency, accessibility, and partition tolerance.

•**Consistency:** Regardless of the data present, any read always displays a new copy of the data.

•**Availability:** every transaction always ends with a scheduled response.

•**Partitioning tolerance:** possibility to also read and write a database even if some parts of it are completely inaccessible. But this CAP theory states that it is impossible to obtain all three properties simultaneously.

•**ACID et BASE :**ACID is an acronym for the four properties of the relational model and BASE is an acronym for the generic properties of the NoSQL ACID model that was introduced previously.

•**BASE:** Mainly available, ultimate soft, consistent case.

•**Basically available:** The database is accessible in all cases.

Chapter1

•**Flexible state:** the database does not need to be consistent

•**Consistent in the end:** a database can contain copies on several servers. These servers may not contain the same data, but at some point the data on each server is refreshed to get the latest version of the data. Businesses still use this basic approach because it offers high accessibility and is quick and straightforward.

.15.4 Advantages[1] :

.15.4.1 **Horizontal scalability:** Instead of relying on horizontal scalability, database managers have long opted for vertical scalability. Indeed, it is difficult to adapt this strategy to a relational database. NoSQL systems are designed to automatically and transparently distribute data across multiple nodes and create a matrix to maintain linear performance during measurement operations. In addition, servers can be added or removed very quickly. Simply put, NoSQL should never be restarted.

.15.4.2 **The economical solution :**NoSQL databases tend to use inexpensive servers to equip the "clusters", while relational DBMS systems tend to use powerful servers at lower cost. In addition, professional database management systems require expensive licenses and NoSQL DBMS are open source and free solutions .

.15.4.3 **Handling Large Amounts of Data:** By processing large amounts of data, it has become virtually impossible for a single relational database server to meet the performance requirements of the business. Today these large amounts of data are no longer a problem for a NoSQL DBMS, even the largest RDBMs cannot compete with a NoSQL database.

.15.4.4 **Less essential database administrator :**While there are many improvements that RDBMS patients have focused on for years when it comes to database management, a sophisticated relational DBMS system requires some expertise to maintain. This is the reason why DBA is generally used by a DBA, so it comes at a certain cost. It would be wrong to say that the existence of a DBA is no longer desirable to manage a NoSQL database. But its tasks will be facilitated in particular thanks to the distribution of data and especially thanks to the simplicity of the datagram. There will always be someone responsible for performance as well as availability when critical data is included .

.15.4.5 **The basic diagrams are not static "shémaless":**Currently, data is more flexible because the structure and type of data can change at any time without necessarily affecting

Chapter1

the application . A NoSQL database is not locked into a single data hierarchy, it is much less restrictive than an SQL database. Thus, NoSQL applications can store data of any format or structure and change the formatting in production. In the end, this means considerable time savings and better reliability.

The interest of NoSQL storage systems lies above all in the software engineering options provided in their designs. Among the most important reasons that led to the implementation of these systems, we find in particular:

- The possibility of using something other than a fixed schema in the form of tables, and all its properties are predefined .

.15.5 Types of NOSQL databases: [8]

.15.5.1 Document stores: The concept of "documents" is the central concept here with documents being equivalent to records in relational databases and tables-like groups. Includes MongoDB and CouchDB document repositories.

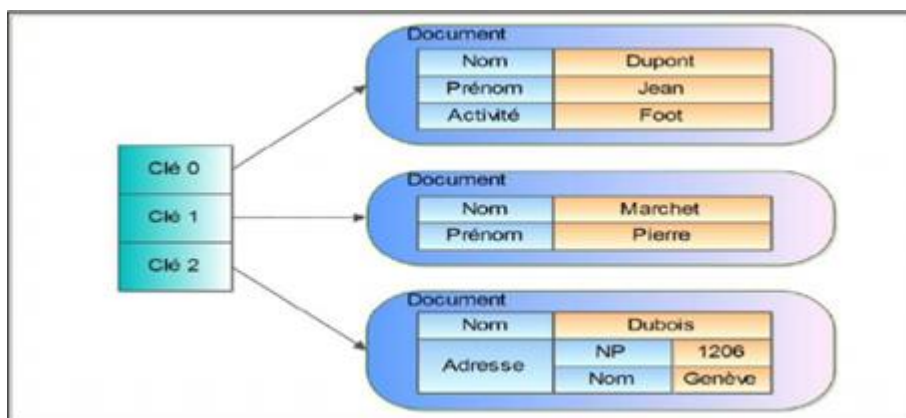


Figure I.1 : Document-oriented databases[8]

.15.5.2 Graph stores: They are used to store information about networks, such as social connections. Chart stores include Neo4J and HyperGraphDB.

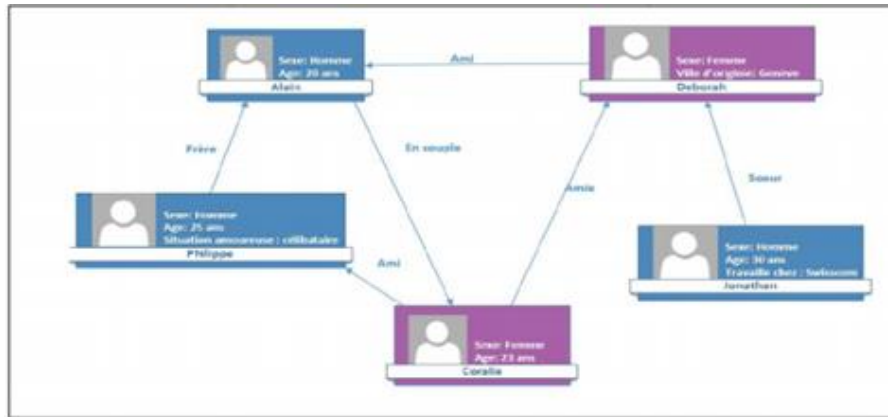
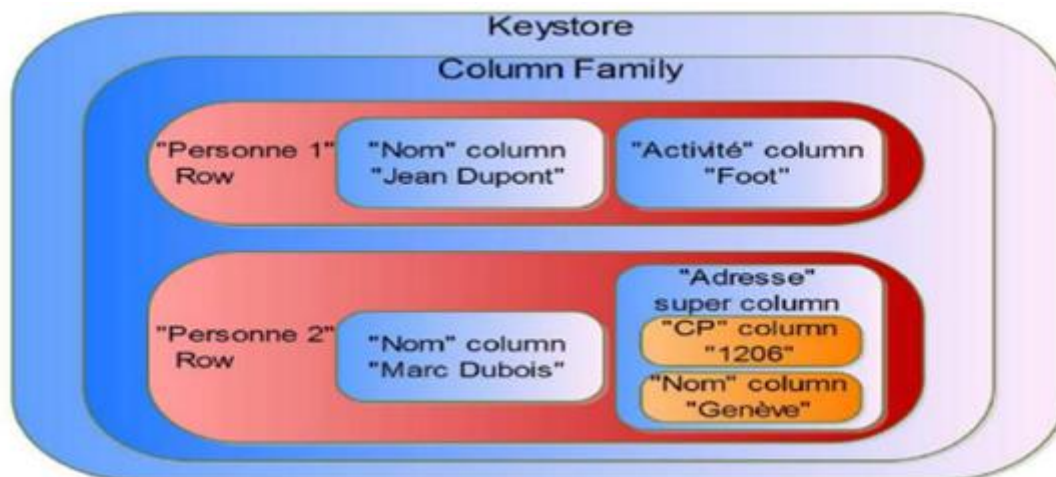


Figure I.2 :Graph-oriented databases[8]

.15.5.3 Column-oriented databases: Each item in the database is stored as an attribute name (or "key"), along with its value. Includes Keyvalue MemcacheDB and Azure Table Storage.



FigureI.3 :Column-oriented databases[8].

.15.5.4Wide column buffers such:

Cassandra and HBase have been optimized for inquiries across large data sets, storing columns of data together, rather than rows. Column stores include Hadoop, Cassandra, and Hypertable.

.15.6 NoSQL Database Properties : [6]

NoSQL databases have the following properties:

- They have higher scalability.
- They use distributed computing.
- They are cost effective.
- They support flexible schema.
- They're able to process both unstructured and semi-structured data.
- There are no complex relationships, such as the ones between tables in an RDBMS.

.16. A comparison between a relational database and a NoSQL database : [9]

Table I.1 : Comparison table between relational database and NOSQL database:

Requirements	SQL	No SQL
ACID compliance (Atomicity, Consistency, Isolation, Durability) for transaction-based applications	Preferred option	Not Suitable Although some databases adhere to CAP theorem (Consistency, Availability, and Partition tolerance), they lack in providing atomicity, and integrity properties
Security and Integrity	Preferred if the security of data is of utmost importance. Strong security features integrated into them, masking encryption, etc. Data accuracy and consistency can be maintained	Preferred if the security of data is not that important (public classified/restricted data). Databases have very few built-in security features, and data integrity is difficult to implement. Data Structure
Data Structure	Preferred option if data is structured, fixed, or changes to the structure are very few or rare and data/records are mainly centralized. Changes to schema may require a change to the entire database, tables, and data updates.	Preferred option if data is unstructured, frequently changing, and data/records are mainly distributed. Data format and data structure changes are very simple to insert without effecting the entire database.
	Prefer databases in	Preferred option if data

Chapter1

Redundancy& Data Normalization	normalized schema where redundancy is very rare	redundancy is expected
Entity Relationship, Consistency of Data	Suitable where the relationship between entities is 'relational,' which must be strictly maintained, and data is strictly consistent all the time.	Suitable where the relationship between entities is 'Hierarchical' in nature. Dynamic or varying and eventual consistency is acceptable
Performance & Availability	Performance of read and write operations may be slower compared to NoSQL, however, the execution of complex queries could be better in a SQL database. Most databases support high availability design.	Better performance (num of read/write operations per sec) for simple queries. Complex queries could have a performance impact. Most databases support high availability design
Query Handling	Good fit for complex queries	Not a good fit for complex queries. Complexity increases in NoSQL query handling
Scalability	Horizontal scalability is a challenge and expensive	Easier to scale
Bigdata	Handling big data is expensive and complex for scalability	Ease of handling big data and low-cost for scalability
Support	Vendor&community support	Mostlycommunity support
RapidDevelopment	Require more time to set up the database. Not suitable for frequent changing data requirements	Preferred with frequent changes to data structure (ambiguity in data that is getting retrieved from sources and getting stored)
Use Cases	Transactional systems, system of records, financial transactions, core banking applications, order management, OLTP,	Content management, personalization, search engines, users profiles management and personalization, data streams, document store,
Examples	MySQL, Oracle, MS-SQL, SQLite, Aurora, Postgres	MongoDB, CouchDB, Redis, Hbase, Cassandra, AWS DynamoDB, Azure Cosmos DB, AWS Neptune, Neo4j, etc.

.17.Some current Solutions:

Solution 1 :

Scaling model - Relational databases are a technology that scales vertically – to add capacity (data storage or I/O capacity) we need a bigger server. The modern approach to application architecture is to scale horizontally, rather than vertically [10]. Instead of buying a bigger server, we use many commodity servers, virtual machines or cloud instances behind a load balancer. In reverse, system capacity can be easily reduced when no longer needed. While scaling horizontally is already common at the application logic tier, the database tier is just starting to use this approach[11].

Solution 2:

Data model - The deployment benefits of NoSQL technology for scaling horizontally frequently get the most attention, but equally important are the benefits granted by a schema-less approach to data management. With a relational database, we must define a schema before adding records to the database. Each record added to the database must strictly comply to this schema and its fixed column names and data types [10]. Bringing changes to the database schema is difficult, especially when it is a partitioned relational database that spreads across multiple servers[11].

Solution 3 :

Relational data model – Besides the need to review the schema every time the data that we want to collect change, this model is characterized by the database normalization process, by which large tables are decomposed into smaller tableslinkedtogether[11].

.18.The Tools Used For Migration :

The following tools in this study: Hadoop, HBase, and Cassandra, which are Java Frameworks for distributed applications and data-heavy management, are of particular interest to researchers.

1.8.1 Hadoop : [12]



Hadoop is a collection of tools and software for developing distributed applications. It's a Java-based open-source software platform based on Google's MapReduce concept and distributed file system (HDFS). It enables distributed applications to be supported by analyzing massive data collections. Hadoop is very useful for indexing and classifying big data volumes, as well as Data Mining and Log Analysis, as well as image processing. It contributes to Google's success in part. The future behemoth creates what inspired core components of Hadoop: MapReduce, Google BigTable, and Google BigFiles in 2001, while it is still in its infancy in the market as a search engine (next generation Google File System). The Hadoop ecosystem is made up of these two parts, a highly sought-after ecosystem at the heart of the Big Data cosmos (Franklin, 2015). Hadoop is made up of two parts: HadoopMapReduce and Hadoop Distributed File System. Hadoop is comprised of the following components:

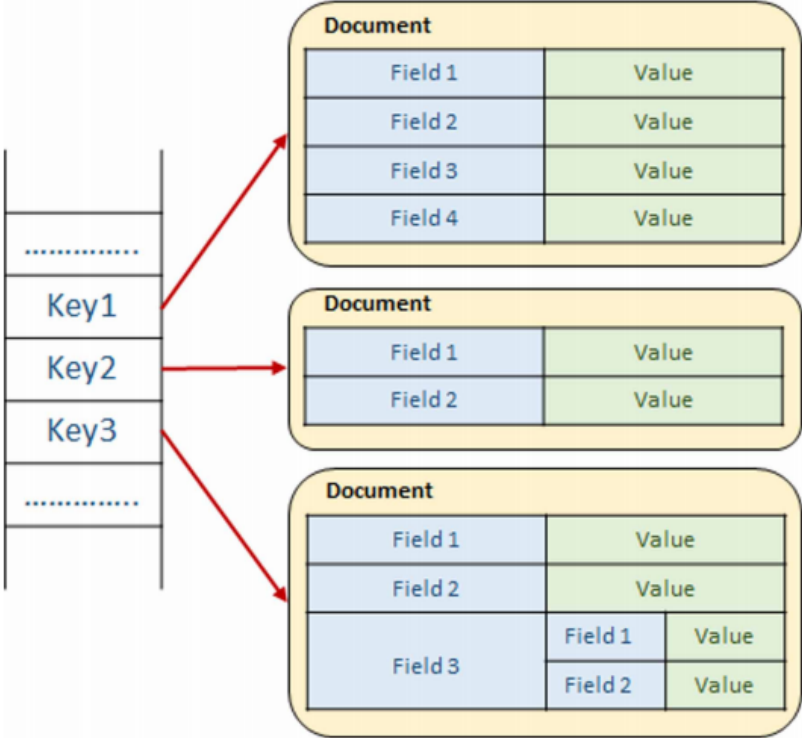


Figure I.4 : Schematic of a database oriented document[12].

- Economical because it reduces expenses by making greater use of resources.
- It's scalable since it uses extra physical resources just when they're needed and in a transparent manner.
- Resilient in the sense that it does not lose data and can continue processing even if a node fails.
- Versatile because it can handle a variety of data kinds and so fulfill the needs of a wide range of users.

Hadoop is built on a filesystem known as HDFS (Hadoop Distributed File System). It enables distributed data storage and high-performance data processing. It employs the MapReduce architecture to distribute operations over numerous nodes, allowing them to run in parallel (see Figure 7). A distributed file system is HDFS. Its great fault tolerance and ability to run on low-cost equipment set it apart from other distributed file systems (Harter et al., 2014). HDFS is meant to assure data security by recording all data written to the Cluster several times. Each data point is written to three distinct nodes by default (see Figure 7). A group of people When compared to typical storage arrays, HDFS has the advantage of providing a very low-cost storage option (White, 2010). [9]

Id	UserName	Email	Departement
1	John	john@exemple.com	IT
2	Michael	Michael@exemple.com	Marketing
3	Eva	Eva@exemple.com	Sales
4	Mark	Mark@exemple.com	Economic

Id	UserName	Email	Departement
1	John	john@exemple.com	IT
2	Michael	Michael@exemple.com	Marketing
3	Eva	Eva@exemple.com	Sales
4	Mark	Mark@exemple.com	Economic

FigureI.5 : Schematic of a column-oriented database.

The architecture of HDFS is master/slave. An HDFS cluster is made up of a single NameNode and a master server that maintains the file system, including file access privileges. Data Nodes are responsible for the storage of data that is allotted to the node on which it is placed. These Nodes respond to the client's reading and writing requests. As illustrated in Figure 8, they also control the formation, deletion, and replication of blocks as instructed by the Name Node (Konstantin et al., 2010).

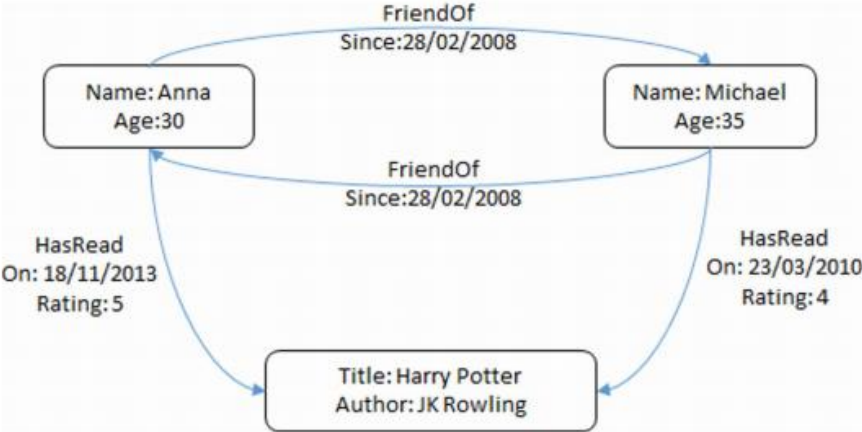


Figure I.6 :Schematic of a graph-oriented database[12].

.18.2.MapReduce : [12]

In the NoSQL world, MapReduce is a distributed programming paradigm that is frequently utilized. It allows parallelizable calculations on a big volume of data to be performed across a large number of devices (Cluster and Grid). There are two essential phases to this technique: The Map function creates tuples (key/value pairs) from the Data collection, and-Reduce is a function that uses these tuples to produce the intended output (see Figure 9).

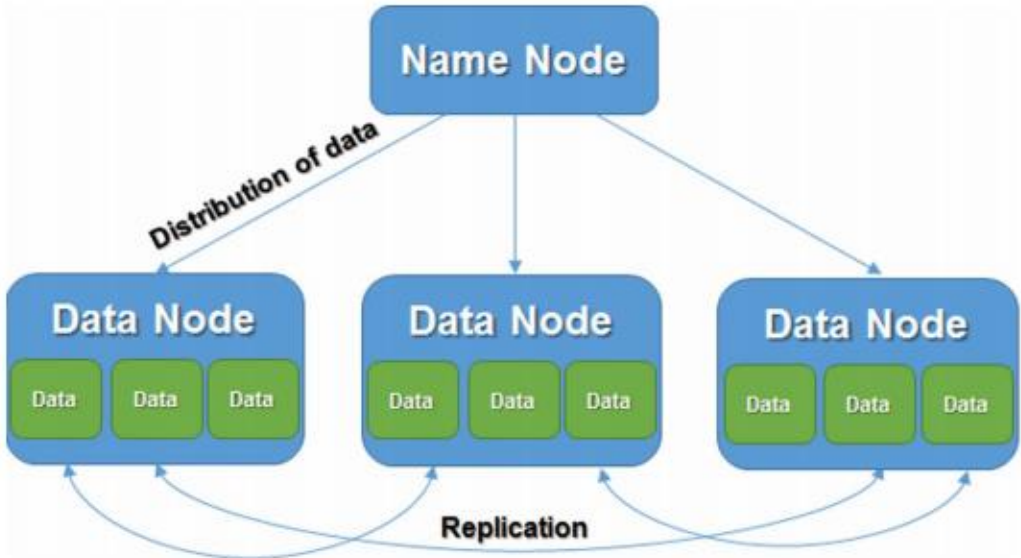


Figure I.7 : HDFS Schème[12].

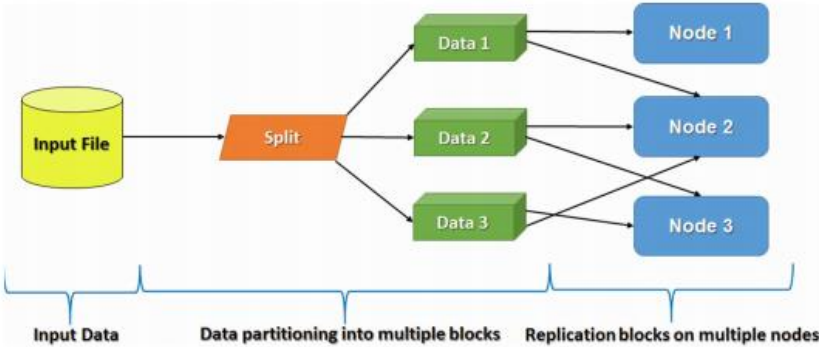


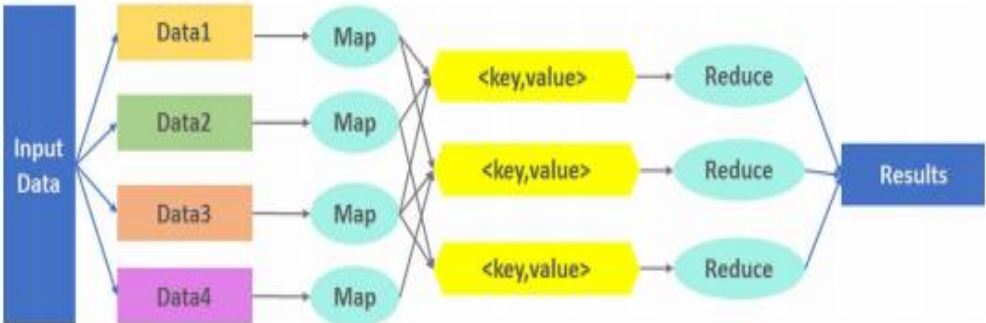
Figure I.8 : Data replication using HDFS[12].

.18.3 Hbase : [12]



HBase is a Java-based non-relational distributed database management solution for huge tables with organized storage. HBase is based on BigTable articles found on Google. BigTable is a column-oriented database management system (Krishnaprasad et al., 2014). Databases with a master/slave design can manage massive volumes of data (billions of rows per table). Hadoop is a distributed architectural framework, and it is a subproject of Hadoop. Although it is not required, the HBase database is typically deployed on the HDFS file system to enable Hadoop distribution.

On November 15, 2010, Mark Zuckerberg revealed that Facebook will be replacing Cassandra with HBase.



FigureI.9 : Functioning of MapReduce[12].

.18.4 MongoDB :



MongoDB is a schema-free document database built in C++ and developed as part of an open-source project led by 10gen Inc, which also provides professional services related to MongoDB. MongoDB's major purpose, according to its creators, is to bridge the gap between quick and scalable key-value stores and feature-rich traditional RDBMS relational database management systems.[13]

.18.5 Cassandra : [12]



Cassandra (Hewitt, 2010), an Apache Foundation project, is a distributed database with horizontal scalability that can hold a big quantity of data.

Cassandra was created by Facebook in 2007. They opted to develop their own database to satisfy their demands because relational databases had reached their limits. After two years of work, they decided to make it available to the public for free, and in 2009, it became an open source project under the Apache Foundation.

Cassandra is built on the foundations of two other databases. The first is BigTable, which was developed by Google and is known for its column-oriented data model and disk persistence method. The second is Amazon's Dynamo, which is known for its distributed design and lack of a master node. Cassandra is a database that can manage a big quantity of data quickly (Kuhlenkamp et al., 2014a). Because of its representation in columns, it allows for more flexible data visualizations. Furthermore, itIn a distributed context, its architecture allows it to develop without issues. It has data replication capabilities as well as the ability to cluster several Cassandra servers. Apache Cassandra, which was based on

Chapter1

Google's BigTable at first, stores data as a sorted series of key/value pairs. Without getting into actual column storage, Cassandra is classified as a column-oriented NoSQL database. The following are the key ideas of Apache Cassandra's data model:

Column :

In the Cassandra data model, a column is the smallest unit. It's a three-part string that includes a name, a value, and a timestamp. This is used to find out when the most recent update was made. The name can have a maximum capacity of 64 KB. The value may hold up to 2 GB of information. The value box can have several values. This is the case with a collection of strings, for example.

Line :

A line is made up of several columns. A key is used to identify it. A single key may store up to 64 KB of data and two billion columns. The primary key might be made up of columns. The number of columns in two separate rows isn't always the same, and the columns aren't always the same.

Column Family :

A logical collection of lines is referred to as a family of columns. A family of columns can be compared to a table in the realm of relational databases. Cassandra differs between two sorts of column families:

- **Static:** the columns are defined when creating or modifying the family of Columns.
- **Dynamic:** the columns are defined when creating or modifying a line.
- A keyspace is a collection of columns that are grouped together as a family. When compared to relational databases, this is a form of schema.

Dede et al. (2014) describe Cassandra's architecture as a distributed and decentralized "Peer-To-Peer" paradigm. At least from the client's perspective, all aspects of a platform are presumed to be similar. Through an internal communication system known as "Gossip," the collection of elements or nodes is kept in a coherent functioning state. The Cassandra architecture aims to provide high availability by combining strong fault tolerance with simple and flexible scaling, all while promising good speed for both writing and reading. Figure 10 depicts the relationship between a Cassandra platform's capabilities and the number of connected nodes.

1.9. Conclusion:

Databases are one of the most important parts of the tech world. The relational structure stands out as a clue and is widespread thanks to its skills and strength resulting from the obstacles posed by this model.

The same principles that are the strength of this model make it less efficient and less responsive in a distributed environment because it prevents data from being distributed without qualification in the nodes that make up a large-scale distributed environment suddenly, and in the face of these obstacles.

Chapter 2:

Approach for Migration

2.1. Introduction :

Chapter2

Many firms have stored their data in relational databases and now want to use the newly developed column-oriented databases. As a result, a procedure or method for migrating from a relational database (RDB) to a NoSQL database must be defined.

In this chapter, we focus on converting a relational database to a column-oriented NoSQL database (BDNOC), Database migration is a procedure that involves all of the components. We will discuss each of the following elements :Source Data Model ,Objective Data Model, Translating the source data model into a target model.

2.2. Source Data Model:

We normally employ the physical data model (PDM), i.e. the relational database script, as the source data schema (PDM) during the migration process, and more precisely during the schema translation step. A logical perspective of how data is represented by a DBMS is to organize it into tables or relationships. The physical view (i.e., how the data is really kept) is not visible to the user and is [14]separate from the logical view. In reality, the relational model's connection does not include cardinals. It is important to return to the conceptual level. The Entity Association model is now presumed to be the source data model that will be submitted for translation to the DBNOC model. Moving from the PDM to the conceptual data model, and then to the Entity-Association model, requires additional steps.

- **The Entity-Association model:**

The entity-association model is a graphical formalism for data modeling that emphasizes the entities, their relationships, and the cardinalities of the relations. It gives a graphical description that enables data models to be expressed as diagrams with entities and relationships. These models are employed in the early stages of computer system design. The Entity-Association model has the following concepts:

An attribute, An entity, An association.

- **Many to many associations:**

Chapter 2

An association or relation R that connects two entities A and B is of the many to many type when numerous instances of entity A can correspond to numerous instances of entity B, and vice versa. It appears as follows.

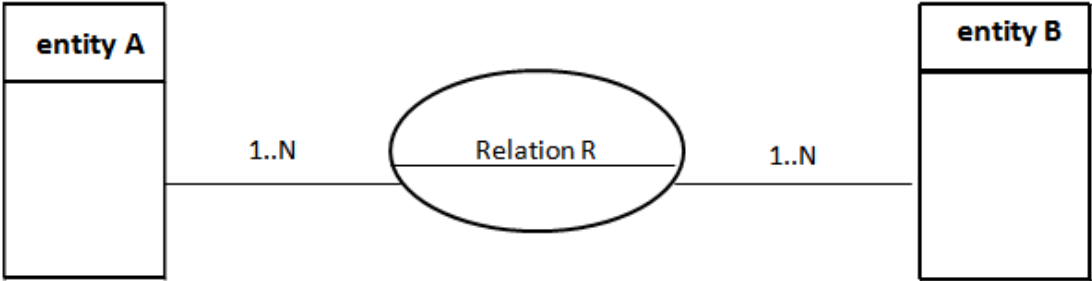


Figure II.1: Relation many to many

- **Many-to-one association:**

When numerous instances of entity B can correspond to a given instance of entity A, an association or relation R connecting two entities A and B is of type one to many or many to one. As indicated in the diagram below, entity B corresponds to one and only one instance of entity A.

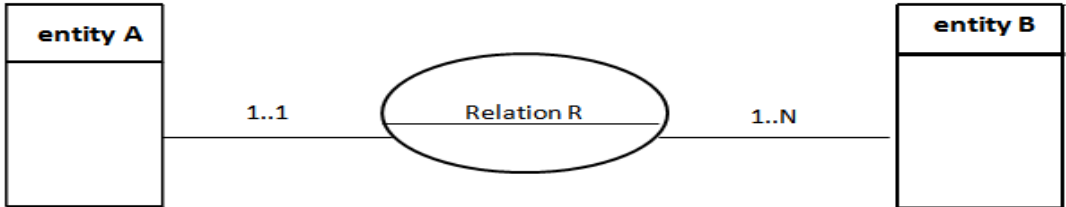
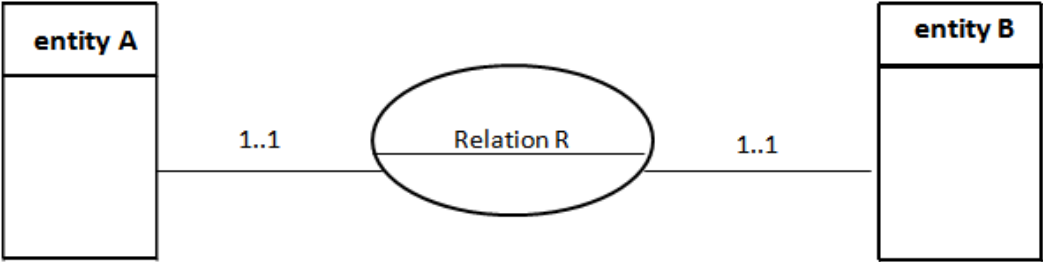


Figure II.2 : Relation many to many

- **One to one associations:**

When a particular instance of entity A corresponds to one and only one instance of entity B and vice versa, the association or relation R is of the one to one kind.



Fig

Figure II.3 : Relation one to one

After rendering the source data model, the target data model must be drawn in order to approach the translation of the relational schema to the DBNOC schema.

2.3. Objective Data Model:

Tables, column families, keys, super-columns, and simple columns are all notions that are used in DBNOC, which is a complex data structure having at least 5 dimensions.

RDBMS is a line-oriented database management system. This means that each column in a row is aggregated and saved one by one .A column-oriented database, on the other hand, prefers to group data by entire columns when storing it. While the columns in a relational database are static and exist for each row, the columns in a column-oriented database are dynamic and exist only for the rows that have values corresponding to each column. To put it another way, the columns are added or removed from a row at any moment. As in RDBMS,

fields containing null are not stored in DBNOC. Column-oriented databases, on the other hand, are built to handle a huge number of columns per row (up to several million). As a result, we see that it's not just about saving entity data, but also about storing relations one-to-many [15].

The many data structures that make up or compose a columnar database will be introduced in the following for a better understanding of DBNOC modeling.

In below, we explain the 5 dimensions of data structure in DBNOC.

2.3.1 DBNOC Data Structures: Table

Tables are seen as Workspaces or domains. In the same group, we distribute families of columns from a table. Before the collection can be launched, certain column families must be defined in the storage-conf.xml configuration file. This data structure is the most detailed. This refers to a set of column families that include all of the application logic. In NoSQL, a table is a program that maintains, processes, and saves data from many sets of columns. As seen in the diagram below, this is the equivalent of a relational database.

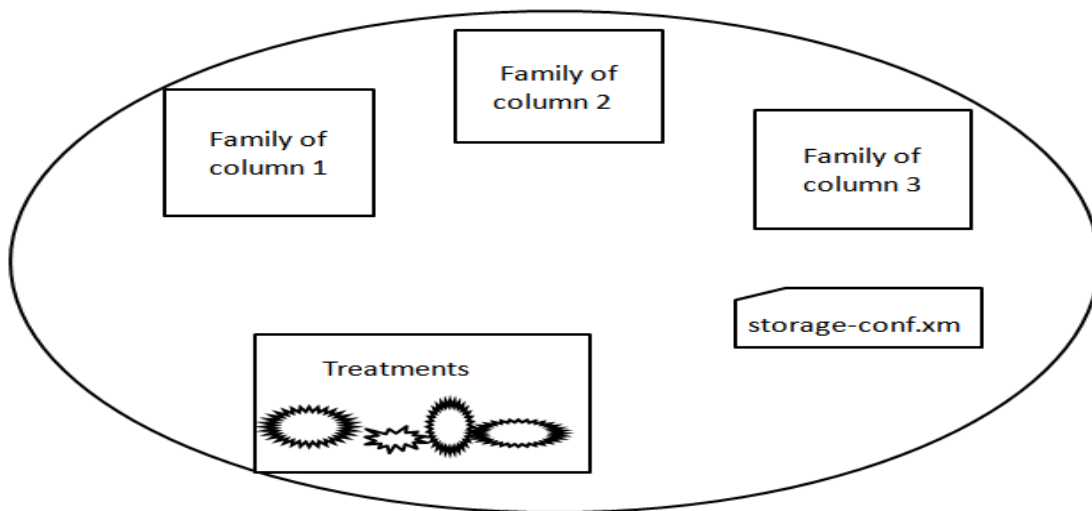


Figure II.4: Table data structure in DBNOC

2.3.2 DBNOC Data Structures: The Key

This is the data line's and record's unique identification. This is a string of characters to generate an index and speed up access to two values. In DBNOC, each column family has its own name, which we can use to query against. It can also be the identifier of an object (a column during the storage).

2.3.3 Data Structures in DBNOC: The Column Family

The column family is the next level of data organization in DBNOC. A table is made up of one or more column groupings. When the collection is launched, the number, name, type and other parameters for column families are set beforehand.

In DBNOC, the table's data is kept in a separate file for each family of columns. Data is saved in rows inside this family (corresponding to the keys). Columns that have linkages

Chapter2

between them must be grouped into the same family of columns since they are accessible at the same time.

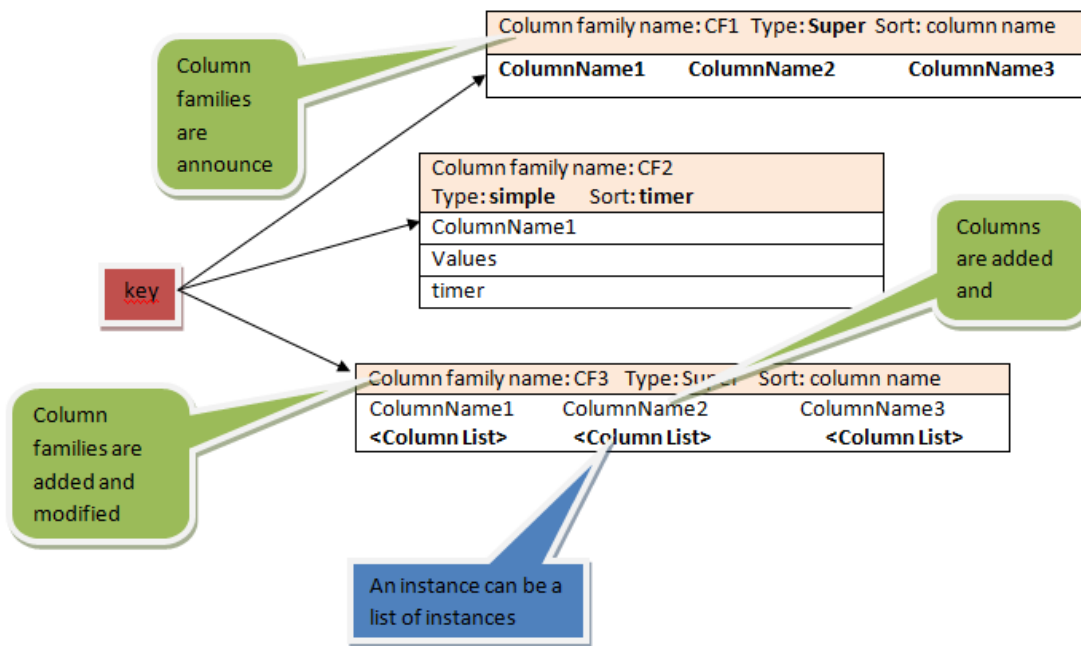


Figure II.5 :Modeling a family of columns

2.3.4 DBNOC Data Structures: Simple Type Columns

Simple columns are structures with a name, value, and timestamp that are defined by the user. The number of columns and their names might be quite extensive, and they can differ from one key to the next. For instance, key (ID) 1 might have two columns or super columns, but key 2 only has one.

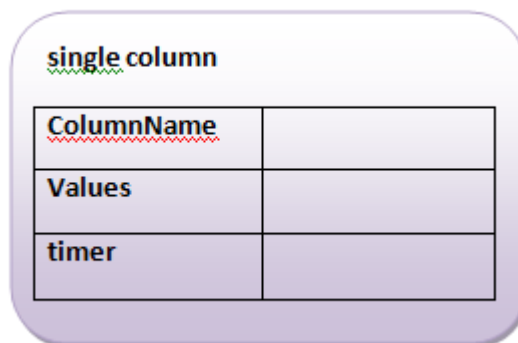


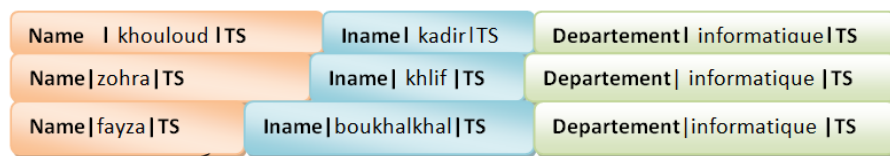
Figure II.6: Sample column of simple type in DBNOC

Chapter2

Below is an example to help you better understand the concepts. On the left is a standard relational database storage diagram, and on the right is its column-oriented storage. The '*' wildcard can be used to execute lookups of the type: table: CF: (key name, *) in the context of basic columns. The relevant collection of groups will be returned (column name, value and timestamp). Sorting columns by name or timestamp is an option.

Id	Name	Iname	departement
1	khouloud	kadir	informatique
2	zohra	khelif	Informatique
3	fayza	boukhalkhal	informatique

Organization of a table in a RDB



Organization in a BDNOC

Structure of a column simple in logic BDNOC

TS: TimeStamps means time marker

Figure II.7: Example of storage representation according to the DBNOC model

2.3.5 DBNOC Data Structures: Super type Columns

They're structures that let divide a column into several sub-columns endlessly. Super columns function similarly to column containers. This gives the form a new dimension by allowing it to store sample lists. The data model that results from all of the foregoing is as follows.

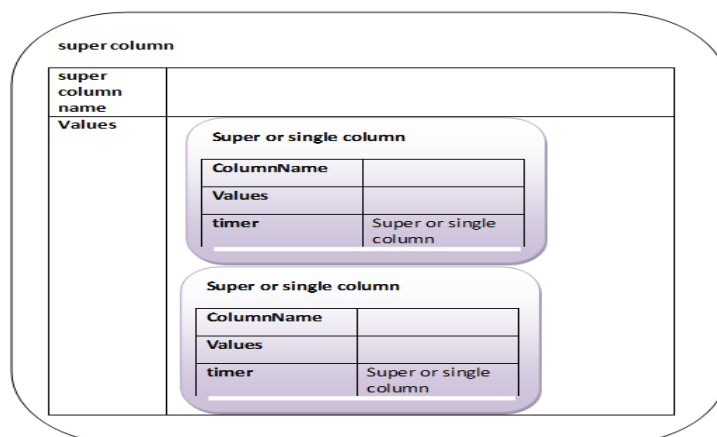


Figure II.8 :Supercolumn modelling in DBNOC logic

2.4 Translating the source data model into a target model:

By implementing a set of translation rules, an S1 data schema may be converted from an RDB database into an equivalent target schema, described as a DBNOC model.

The conversion of the source schema to the destination schema is separated into two phases. The goal of the first phase is to get the source database's conceptual diagram. This is the entity association (E-A) model.

In the second phase, the objective conceptual diagram is created from the conceptual diagram acquired from the first phase.

The source schema, on the other hand, may be translated straight into the destination schema without the need for intermediary representation. To improve results and exploit target database ideas, an experienced user or tool may be requested to give missing semantics.

2.4.1 Entity translation:

Each entity attribute will be translated in a column family. We note that in the column family, any number of columns can be present.

This is an example of an entity translation from RDB to DBNOC.

users
-id
-Name
-lname
-departement

Chapter2

tableII.1 : DBNOC Entity Model

Users column family		
Values	Name column	
	Values	
	TS	
	Iname column	
	Values	
	TS	
	departement	
	Values	
	TS	
Timer(TS)		

2.4.2 Translating multiple to one association:

The association of numerous records from one table in database administration indicates that numerous records from table A are linked to a single value from table B. The following is an example of Entity Association Model for a situation where a client past an order or request in market:

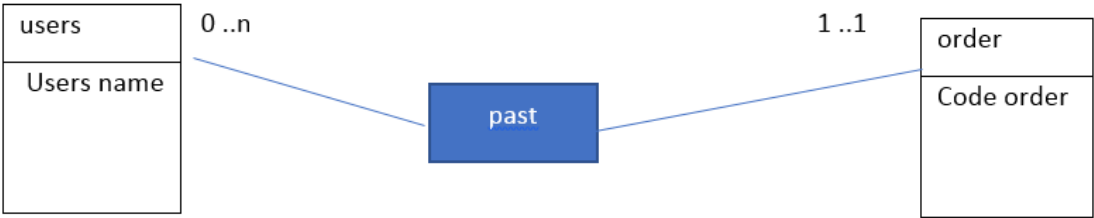


Figure II.9 : Example of a relationship one to several

The DBNOC storage organizes data such that all orders or requests of client C may be retrieved in the same place. If we reach a client C, we will reach all his requests. The following is an example of a data storage method:

Chapter2

Table II.2 : Example data representation of a many-to-one association

User key....	
User key A	<div style="display: flex; justify-content: space-around; align-items: center;"> <div style="border: 1px solid black; border-radius: 10px; padding: 5px; margin: 5px;">Ordre 3</div> <div style="border: 1px solid black; border-radius: 10px; padding: 5px; margin: 5px;">Ordre 100</div> </div> <div style="display: flex; justify-content: space-around; align-items: center; margin-top: 10px;"> <div style="border: 1px solid black; border-radius: 10px; padding: 5px; margin: 5px;">Ordre 120</div> <div style="border: 1px solid black; border-radius: 10px; padding: 5px; margin: 5px;">Ordre 30</div> </div>
User key.....	

For a many-to-one association, the column-oriented data model is as follows:

Table II.3 : Data model for a many-to-one association

Column family: user							
Value	<div style="border: 1px solid black; padding: 5px;"> Column: attr1 Column: attr2 <table border="1" style="margin: 5px auto; width: 80%;"> <tr> <td style="text-align: center;">Super Column:</td> <td></td> </tr> <tr> <td style="text-align: center;">Value</td> <td></td> </tr> <tr> <td style="text-align: center;">timer</td> <td></td> </tr> </table> </div>	Super Column:		Value		timer	
Super Column:							
Value							
timer							
timer							

2.4.3 Translating a many-to-many association:

A maximum of three sets of columns occur from translating a many-to-many association between two entities A and B that are bound by a R association.

Entities A and B will yield two families of columns, FA and FB, each of which will include the characteristics of entities A and B.

It will also be added to both column families FA and FB, the super-column SB and SA, which will include a list of items in Column family FB and FA. In terms of R relation or association translation, it will eventually give rise to a third family of FR columns, with R characteristics as columns.

This is demonstrated in the example below:

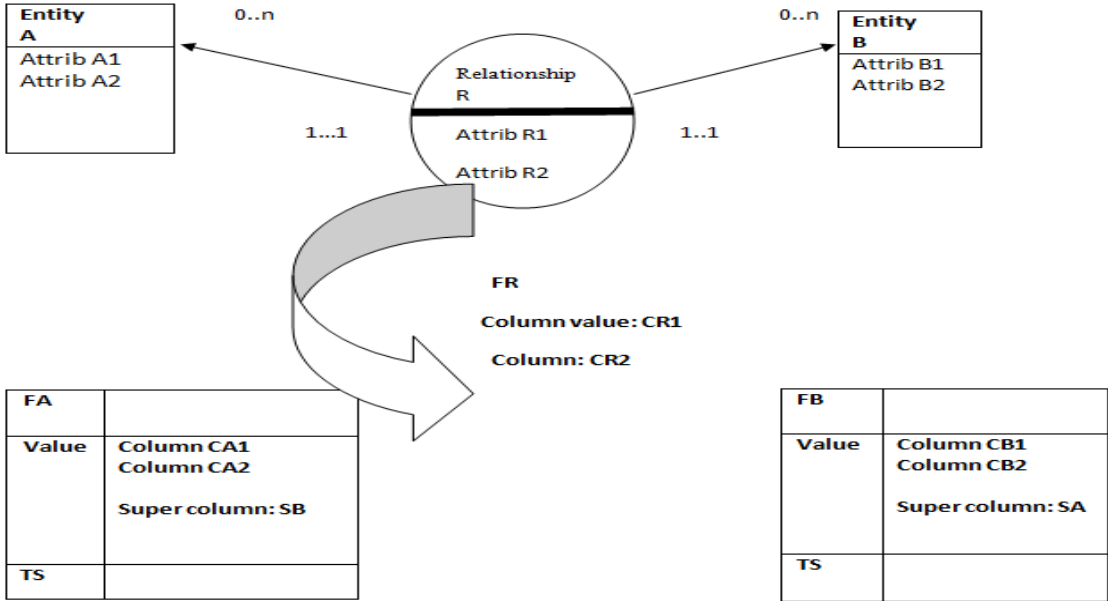


Figure II.10: Translation of the many-to-many relationship.

2.5 Convert data from RDB to DBNOC:

Moving from source data to target data means that data stored as tuples in a RDB are converted to complex structures in NoSQL column-oriented databases.

The data conversion stage must contain the following steps:

- Selecting the fields from the source database;
- Extracting the data from the source database;
- Processing or enriching the data;
- Injecting the data into the destination database.

The diagram below illustrates the three main steps of the data transformation process:

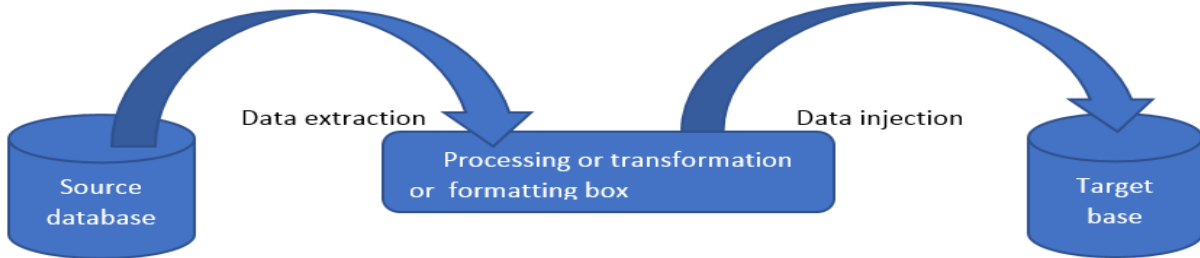


Figure II.11 : Conversion from RDB to DBNOC

2.6.Key correspondence table:

It is a conceptual tool that makes switching between source and target databases easier. It also allows switch from an RDB key to an DBNOC equivalent and back. A key mapping table is an array of structures.

The element structures of a key mapping table array consist of fields, one of which corresponds to a reference to an array. All entries in this field related to non-permanent categories are blank. Refers to a table with two columns, the first column contains the starting keys of the specified table and the second column of the corresponding new key, the mapping master table.

The figure below shows the creation of the main map table during translation from the source data model for the exact target data model:

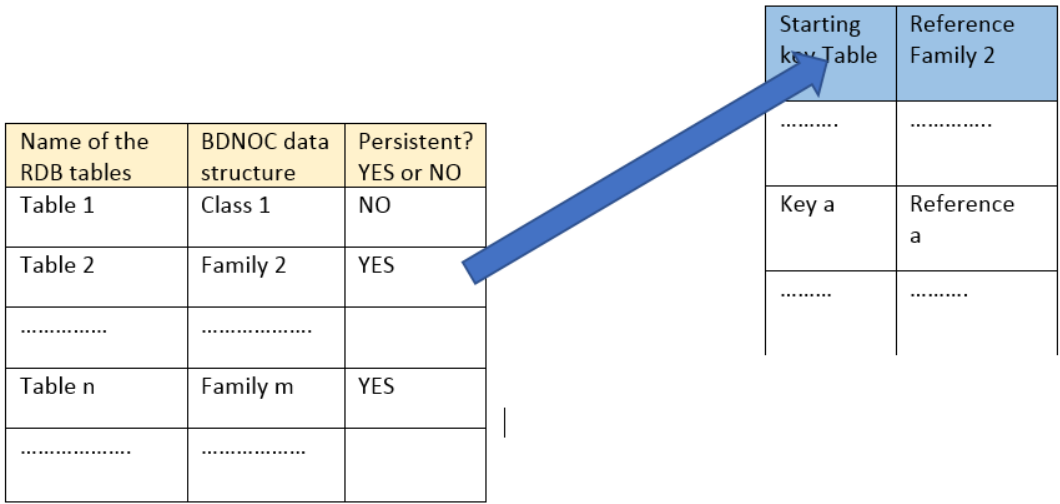


Figure II.12: The main correspondence table

2.7. Conclusion:

In the chapter, we came to the problems facing the MySQL database, and from them we suggested methods and models for migrating to the NoSQL database, which is the MongoDB program.

Chapter3 :

Implementation

3.1. Introduction :

We introduce in this chapter MongoDB and its features, and look at how to download and start its server. In addition, we will show an application example showing the method of migrating from a relational database to a NoSQL database.

3.2. Definition of the MongoDB :

MongoDB is a schema-free document database built in C++ and developed as part of an open-source project led by 10gen Inc, which also provides professional services related to MongoDB. MongoDB's major purpose, according to its creators, is to bridge the gap between quick and scalable key-value stores and feature-rich traditional RDBMS[16].

3.3. Definition of the Mysql :

MySQL is a SQL (Structured Query Language)-based relational database management system. Data warehousing, e-commerce, and logging applications are just a few of the uses for the program.[17]

3.4.Installing single node MongoDB:

In our work, we will look at installing MongoDB in the standalone mode. This is the simplest and quickest way to start a MongoDB server, but it is seldom used for production use cases. However, this is the most common way to start the server for development purposes.

3.4.1 Getting ready:

We have downloaded the MongoDB binaries from the download site, extracted it, and have the resulting bin directory in the operating system's path variable. The binaries can be downloaded from:

<http://www.mongodb.org/downloads> after selecting your host operating system.

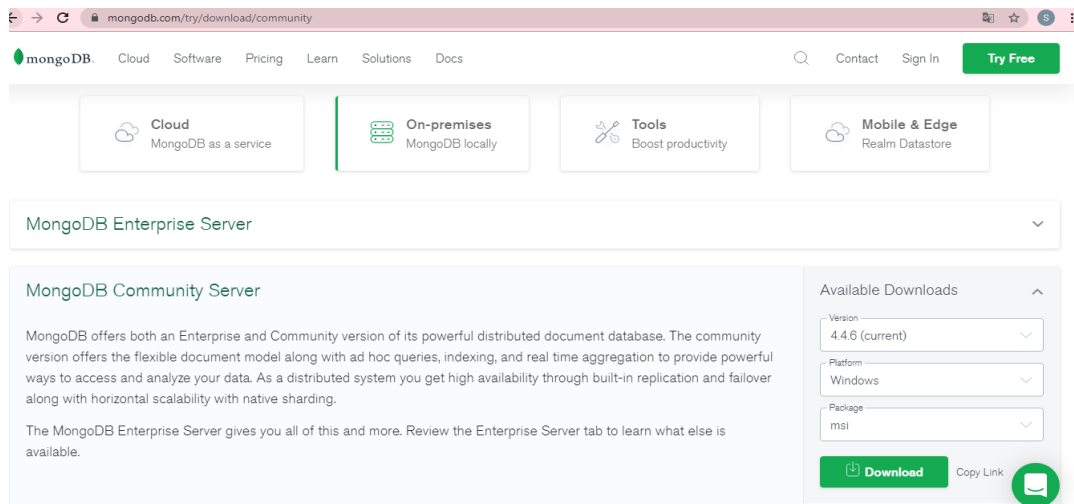


Figure III.1 Dowling of MongoDB

Create the directory, /data/mongo/db. This will be our database directory, and it needs to have permission to write to it by the mongod (the mongo server process) process.

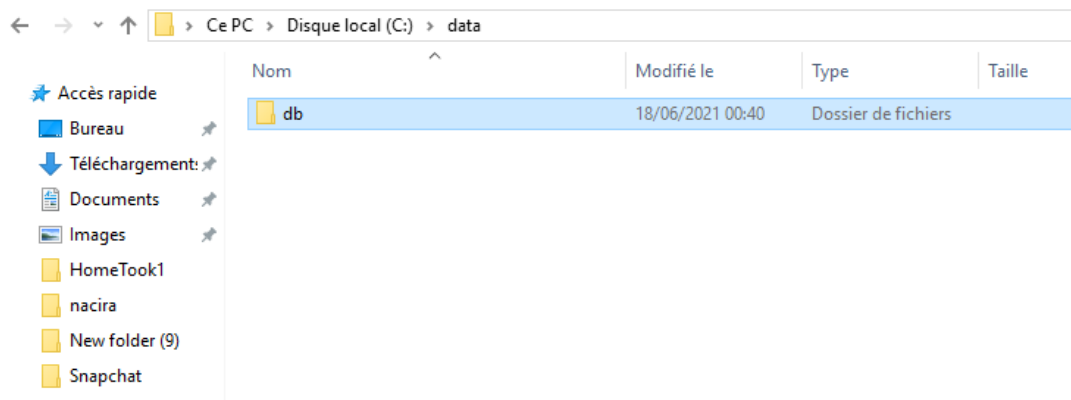


Figure III.2 : Dowling of MongoDB

* Mongod is responsible for creating and managing the MongoDB database.

* Mongo is a MongoDB command line compiler, used to manipulate the MongoDB database and its objects.

* MongoDBCompassis used to save all databases as well as collections in CSV and JSON format.

-The following image shows what we mentioned above:

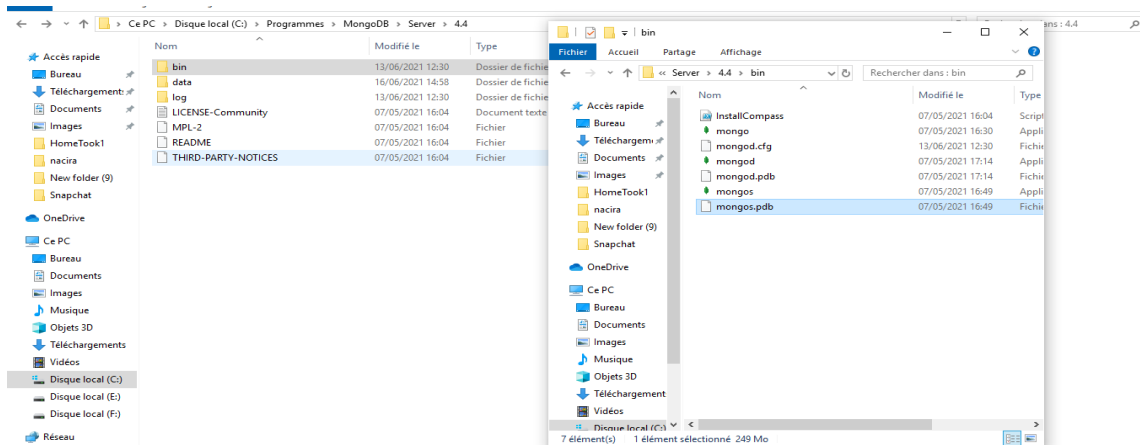


Figure III.3 :the directory of MongoDB .

3.5. Starting the MongoDB Client:

Run an administrator for data management Mongod

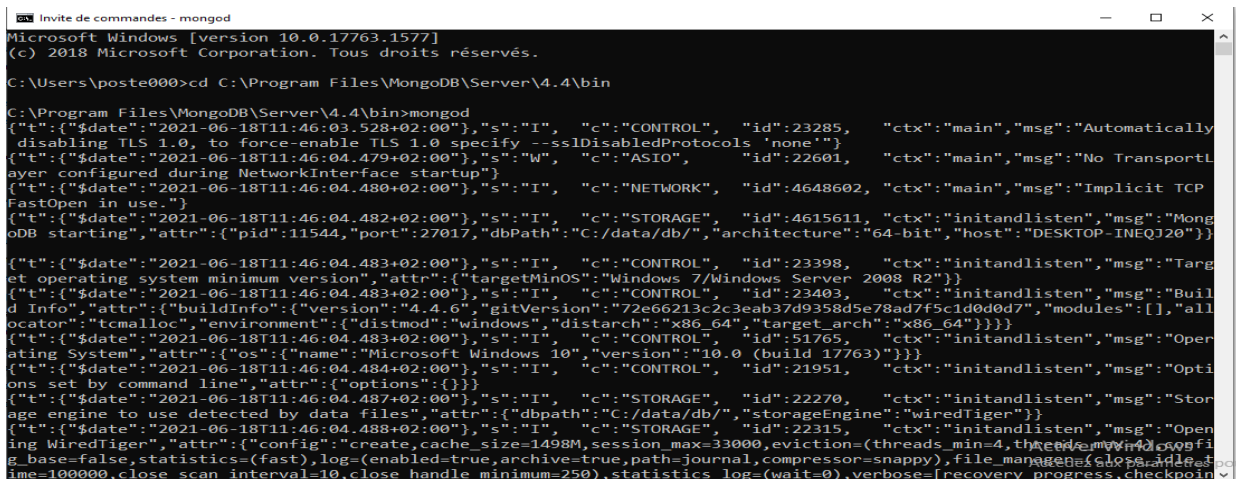
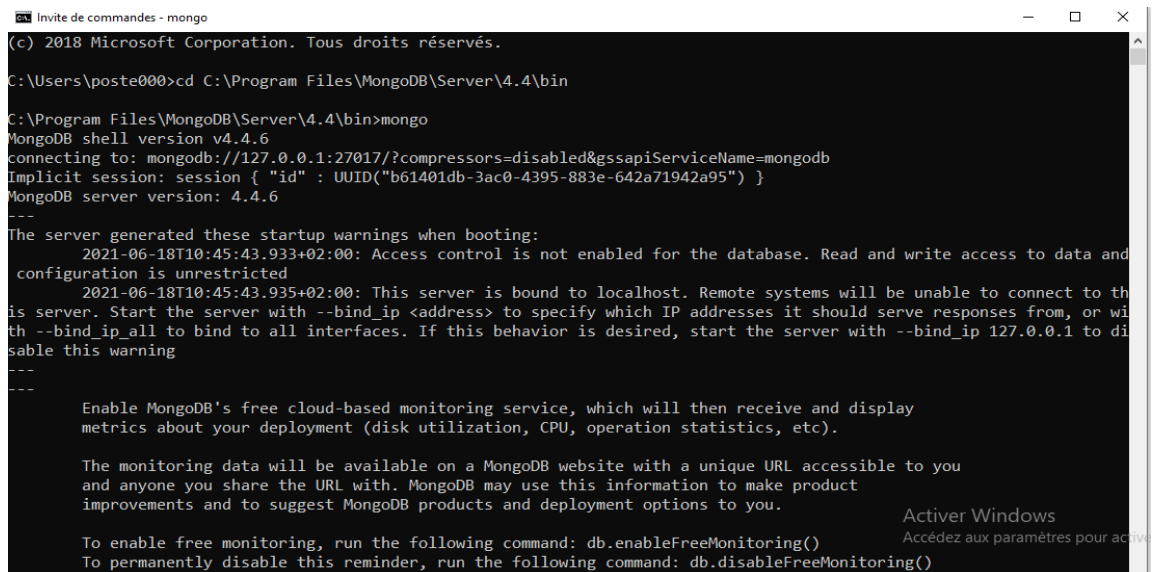


Figure III.4 :the Run of Mongod .

Activate the Mongo command line interpreter



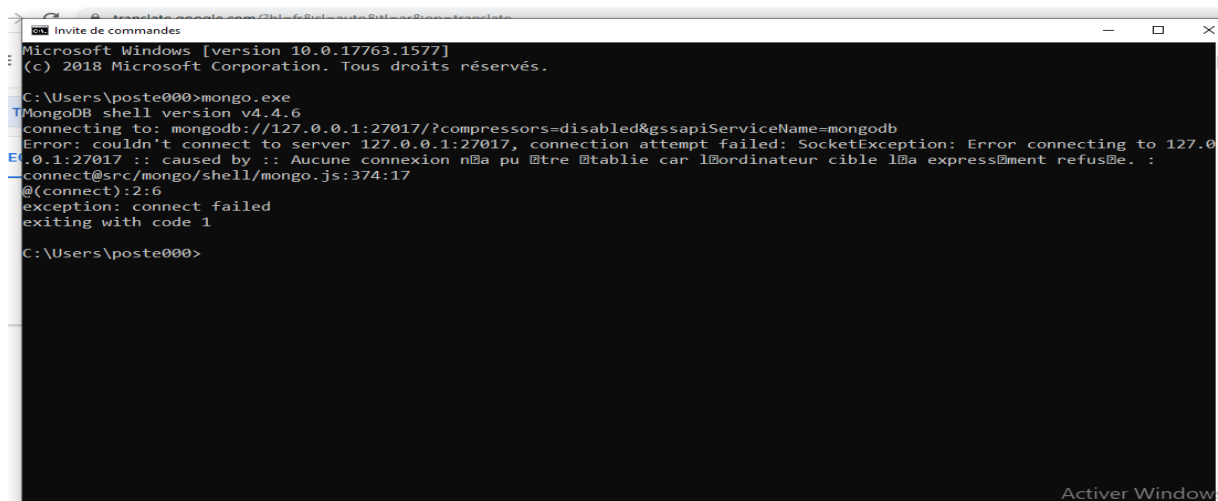
```
Invite de commandes - mongo
(c) 2018 Microsoft Corporation. Tous droits réservés.
C:\Users\poste000>cd C:\Program Files\MongoDB\Server\4.4\bin
C:\Program Files\MongoDB\Server\4.4\bin>mongo
MongoDB shell version v4.4.6
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongod
Implicit session: session { "id" : UUID("b61401db-3ac0-4395-883e-642a71942a95") }
MongoDB server version: 4.4.6
---
The server generated these startup warnings when booting:
  2021-06-18T10:45:43.933+02:00: Access control is not enabled for the database. Read and write access to data and
configuration is unrestricted
  2021-06-18T10:45:43.935+02:00: This server is bound to localhost. Remote systems will be unable to connect to th
is server. Start the server with --bind_ip <address> to specify which IP addresses it should serve responses from, or wi
th --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to di
sable this warning
---
---
  Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

  The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

  To enable free monitoring, run the following command: db.enableFreeMonitoring()
  To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
Activer Windows
Accédez aux paramètres pour active
```

Figure III.5 :the Run of Mongo .

Activate mongod by running Command Prompt and write the instruction mongo.exe



```
Invite de commandes
Microsoft Windows [version 10.0.17763.1577]
(c) 2018 Microsoft Corporation. Tous droits réservés.
C:\Users\poste000>mongo.exe
MongoDB shell version v4.4.6
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongod
Error: couldn't connect to server 127.0.0.1:27017, connection attempt failed: SocketException: Error connecting to 127.0
0.1:27017 :: caused by :: Aucune connexion n'a pu être établie car l'ordinateur cible l'expressément refuse. :
connect@src/mongo/shell/mongo.js:374:17
@(connect):2:6
exception: connect failed
exiting with code 1
C:\Users\poste000>
```

Figure III.6 :Activationmongodb.

Activate MongoDB Compass by copying the link : <mongodb://127.0.0.1:27017>

-And press the button connect

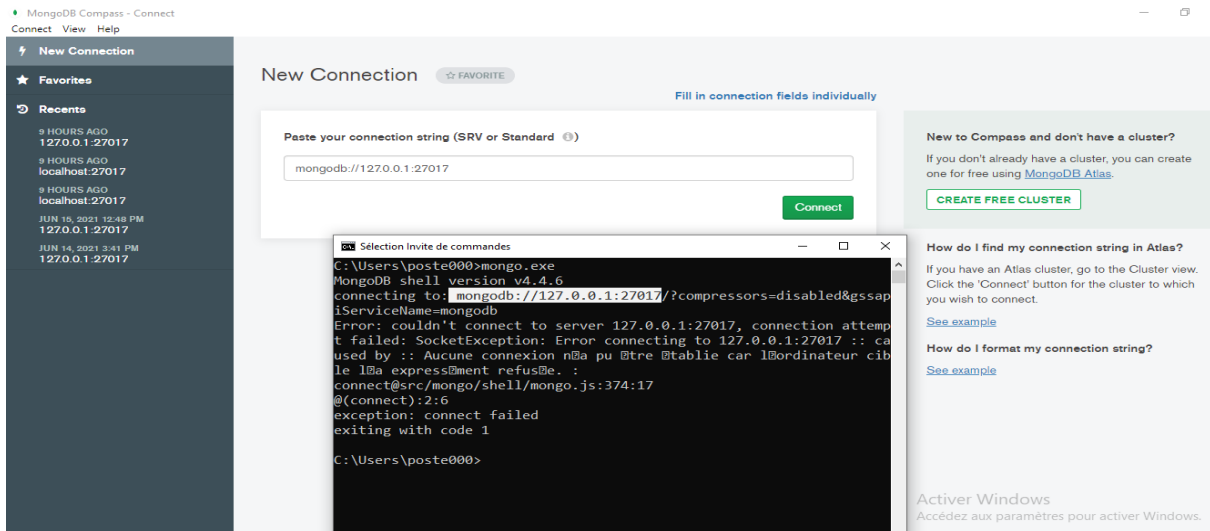


Figure III.7:ActivationMongoDBCompass.

3.6. Our work steps :

For the purpose of studying our project, we suggest this example: It is a website for registering student trainees in a training course at the university. The college is the only one that runs it. This site contains the names of the subjects taught in the college, and is used to request compensatory classes by the professor or student.

a. Log in to Mysql by entering its password:

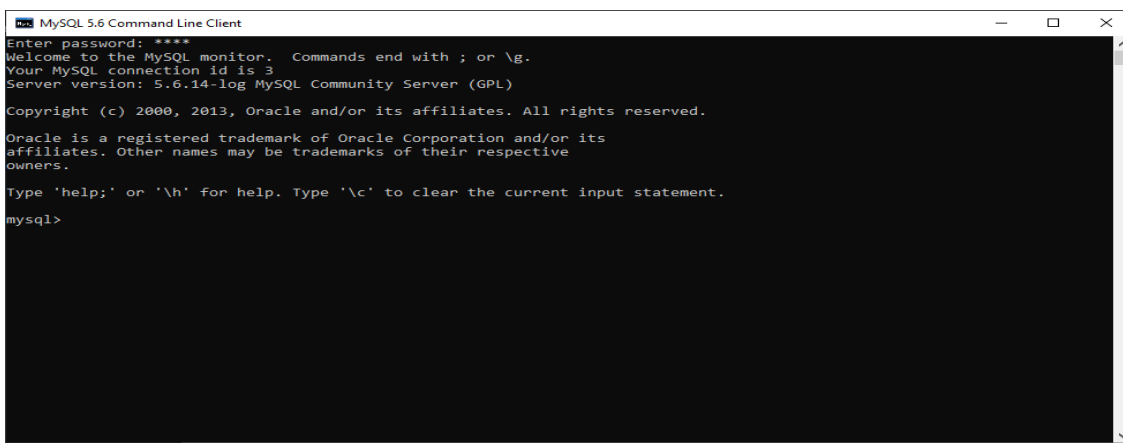


Figure III.8:Log in to Mysql.

d. We created the **exomp1** relational database, and included each of the following tables:

Department ,Hissa , Modele , Users , Requist.[18]

-And insert the data into each table.

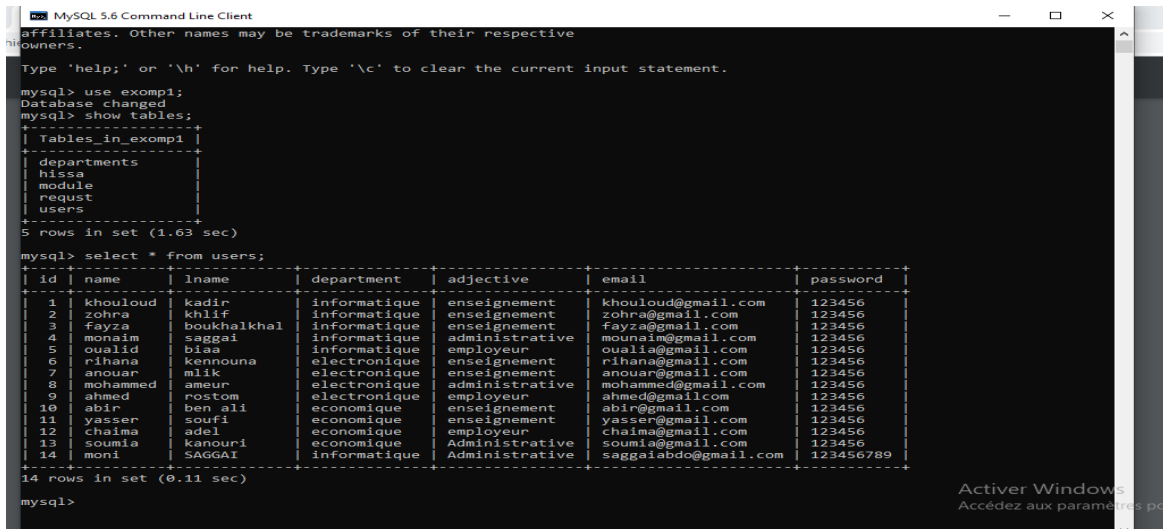


Figure III.9:insert the data into each table.

3.7. Migration steps: After creating the relational database to be converted to the NoSQL database, to convert it, we took the following steps:

- Collect data from mysql.
- Convert it to a CSV file using the link https://www.convertcsv.com/sql-to-csv.htm?fbclid=IwAR11dTArSfYciSwzVnKHT0_5AxH3oT4JRU0E9h4NV4STa_OuB1KkuluWbWM
- Create a database in MongoCompass and insert CSV files into a collection.

The image shows the creation of the exomp1 database and the tra_course collection.

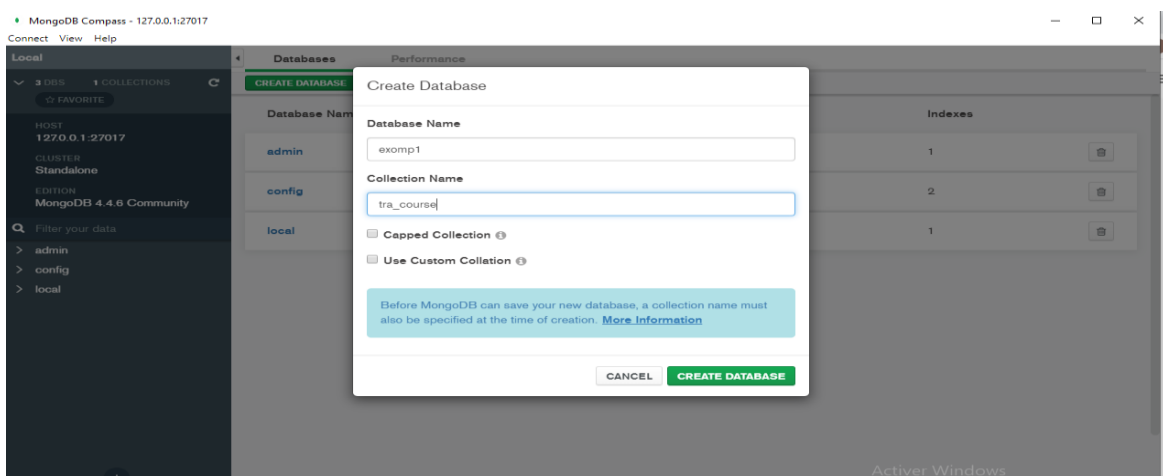


Figure III.10:Create a database in MongoCompass.

Chapter3

The image shows what the exomp1 database looks like after it was created :

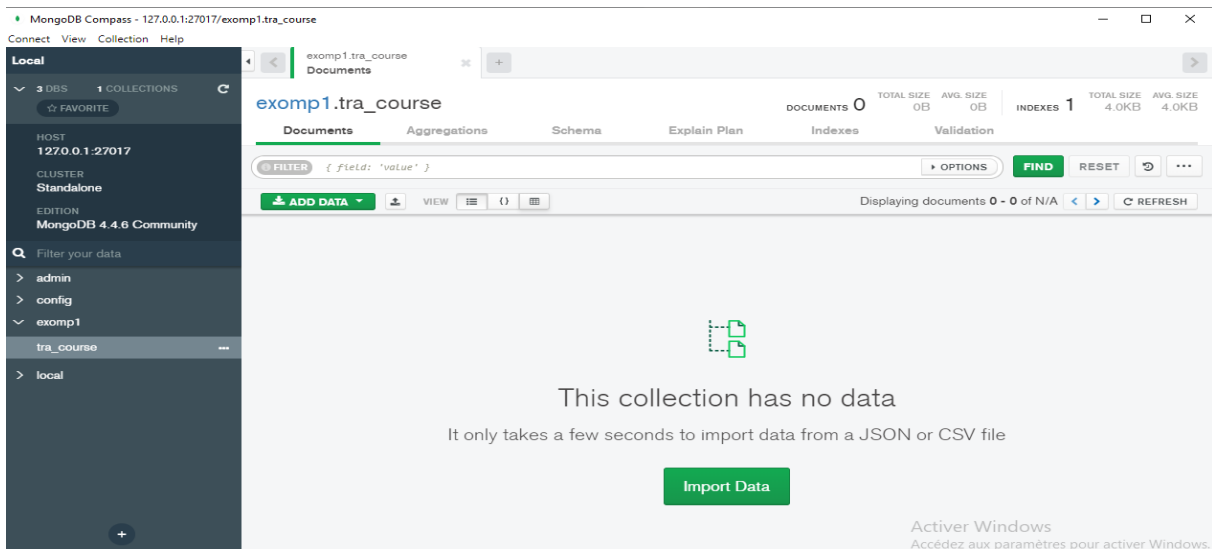


Figure III.11:database looks like after it was created.

The image shows adding the table to the tra_course collection using the ADD DATA button and then pressing BROWSE to add the CSV file.

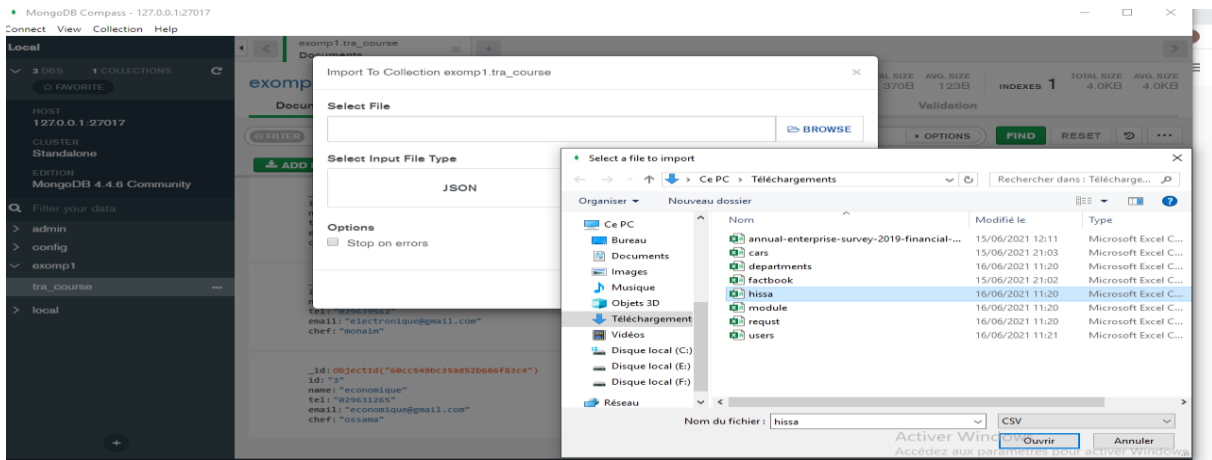


Figure III.12:Adding the ADD DATA to the CSV file.

Chapter3

-After downloading the CSV file, we press import.

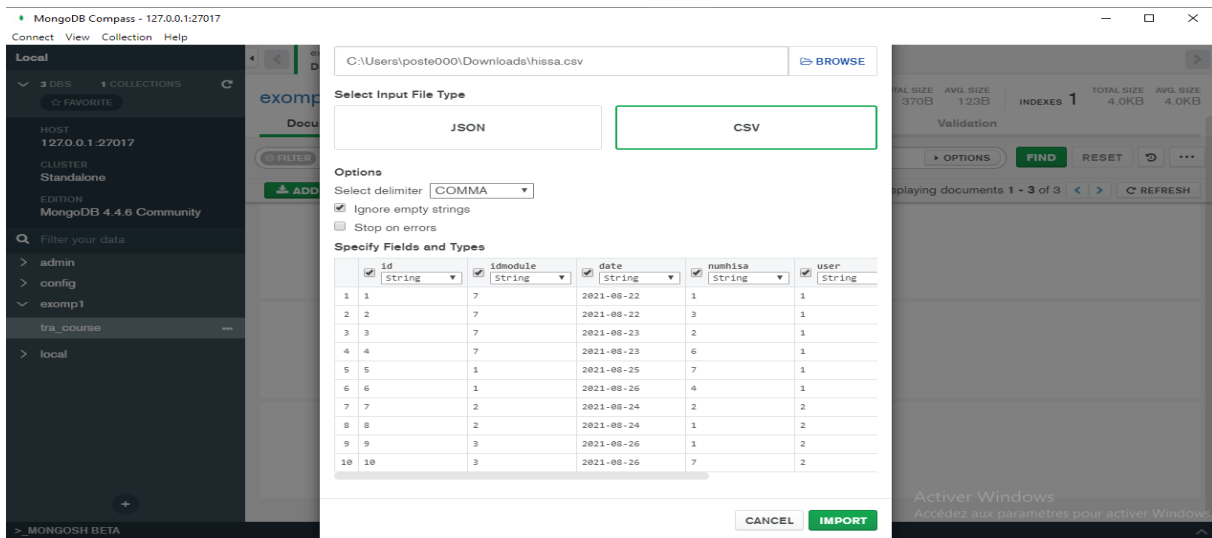


Figure III.12:After downloading the CSV file.

Finally, form the relational database after the migration to NoSQL.

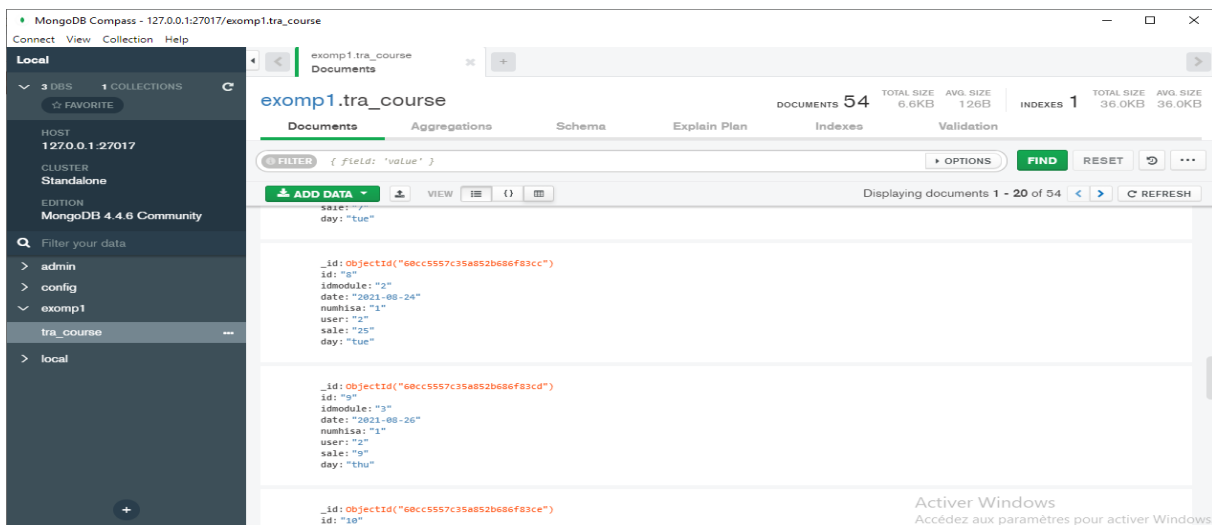


Figure III.13: the database after the migration to NoSQL

3.8 Conclusion:

At the end of this chapter, we have come to the steps of migrating from relational database to NoSQL database. In addition, we got results and showed us that mongodb solution is better performing and responsive than MySQL. And we concluded that the NoSQL MongoDB database is easy to control.

General Conclusion :

In the near future, it is likely that big data will be used in most fields a lot to address the problems faced by database management in large scale environments. Hence, the NoSQL database is a powerful technology which is used in high data environments such as Google, yahoo, instagram.etc. and search engines in particular. They need storage, power and processing speed for such huge data.

NoSQL is the ideal solution for solving the problem of big data, since it has various advantages mainly for big data concepts.

The aim of this work is to prove the competence and power of NoSQL using MongoDB. It is the proposed approach to migrate the relational database for the formative course at the university to the NoSQL database.

Given a distributed context, and the mongodb software Our solution used data export, and data was imported into mongodb from CSV files.

Finally, this work will be a quantum leap in the spread of NoSQL technology and the generalization of its use, and we conclude that the above-mentioned goal has been achieved, and the transition stages can be developed in the future.

Library :

[1]. BACHIR Ahmed .Migration de la base de données des passeports biométriques sous Oracle vers une base de données NoSQL MongoDB.2016.

[2]. Bhojaraju Gunjal. Database Management: Concepts and Design .Conference Paper· December 2003. <https://www.researchgate.net/publication/257298522>

[3].To Catherine Maree Arnold (1981-2010).www.it-ebooks.info

[4]. Mathieu Roger, synthèse d'étude et projets d'intergiciels: bases NoSQL

[5].<https://searchdatamanagement.techtarget.com/definition/relational-database>

[6].<https://docs.rackspace.com/support/how-to/properties-of-rdbmss-and-nosql-databases/>

[7].Gestion de données : du modèle relationnel vers le modèleNoSQL

[8]. AMARA Manel Warda.Etude comparative des bases de données NoSQL. Etude comparative des bases de données NoSQL.2015.

[9].<https://dzone.com/articles/relational-vs-nosql-databases-and-rdbms-db-tonosq?fbclid=IwAR3I8Xhu4RJoo20xjpTOjJCYxTZIiFoJr7qJvfqCosXZ27y4r9y87Y-5cvk>.

[10]. MongoDB, NoSQL Databases Explained,

(<http://www.mongodb.com/nosqlexplained>) accessed on March, 31st , 2014

[11]. Database Systems Journal vol. V, no. 2/2014 . Cristina BĂZĂR, Cosmin Sebastian IOSIF University of Economic Studies, Bucharest, Romania cristina.bazara@gmail.com, iosifsebastian@yahoo.com

[12]. Samah Bouamama. Article *in* International Journal of Knowledge-Based Organizations · June 2018. <https://www.researchgate.net/publication/325608248>. Science and Media (CSM).

[13]. NoSQLDatabases.ChristofStrauch(cs134@hdm-stuttgart.de).Selected Topics on Software-TechnologyUltra-Large Scale Sites.Prof. Walter Kriha.Computer Science and Media (CSM).Hochschule der Medien, Stuttgart(Stuttgart Media University).

[14] AbdelsalamAmragaMaatuk, migrating relational databases into object-based and xml databases

[15] <http://www.portailsig.org/content/le-nosql-dans-le-domaine-geospatial-approche-preliminaire-enpython-avec-simplegeo>, Fevrier 2012

[16].Selected Topics on Software-Technology Ultra-Large Scale Sites. Computer

[17]. <https://www.123-reg.co.uk/support/servers/what-is-mysql-and-why-do-i-need-it/>

[18]. La base de données a été utilisée par les étudiants.Kasdi Merbah Ouargla.2020.