Document-Oriented NoSQL Databases

Performance analysis of MongoDB and MySQL using PHP

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering

In

Software Engineering

Submitted By

Dhulfiqar Hazim

801231007

Under the supervision of:

Ms. Karamjit Kaur

Lecturer

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

PATIALA – 147004

July 2014

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, **"Document-Oriented NoSQL Databases"**, in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Software Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Ms. Karamjit Kaur* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Dhulfiqar Hazim
ME (Software Engineering)
801231007

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Karamjit Kaur
Lecturer
CSED

Countersigned by

(Dr. Deepak Garg)
Head, CSED
Thapar University
Patiala

(Dr. S. K. Mohapatra)
Dean (Academic Affairs)
Thapar University
Patiala

Acknowledgement

I am very thankful to GOD to make such an inspiration, supporting and encouraging guide *Ms. Karamjit Kaur,* Lecturer in Computer Science & Engineering Department, she helps me in thesis and in my life too, she makes me know how to be a real teacher. And her husband *Mr. Gurpreet Singh* for supporting me in my thesis work.

I am thankful to *Dr. Deepak Garg*, Head of Computer Science & Engineering department Thapar University Patiala, for providing us adequate environment, facility for carrying thesis work.

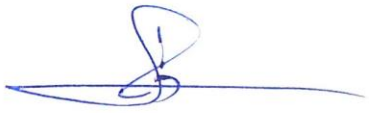
I am thankful to *Dr. Seema Bawa* and *Dr. Maninder Singh* for teaching and encouraging me in the last two years.

I would like to thank all my friends in Thapar that helping me a lot for finishing my master degree.

I am deeply indebted to my father *Dr. Hazim* and my mother *Mrs. Hana*, who always care, support and encourage me for thinking beyond limits and they are the main reason I am continuing my degree in India. And my sister *Dr. Dalia* and my brother *Dr. Ahmed* who always care and supporting me.

I would also like to express my appreciation to my fiancée *Mariam* for her caring with love with all this distance. And my best friend *Ehab Safaa* that I felt like he was with me all my studying time in India and for helping me whenever I need.

Dhulfiqar Hazim



Abstract

Huge amounts of data that is increasing day by day cannot be managed easily by relational databases because of low scalability provided by these databases. The storage technology is still not capable enough for the performance and scalability that is needed, after 2005 NoSQL databases have come and start solving the problems that relational databases was facing before.

NoSQL is a type of database that under non-relational databases. There are four types of NoSQL databases, these types are (Key Value Store– Column Store – Document databases – Graph databases) each one of these databases has different presenting, advantage, disadvantage, and features.

Non-relational databases giving a significant change of how enterprise applications built. The questions of whether non-relational databases are the right choice or stay with the old relational databases for applications and web development, from where NoSQL came from, how they are represented , and what are the types of relational and non-relational databases, these questions are going to be explained in this thesis.

The objectives of this thesis are to show that the need of NoSQL databases became necessary with the time, second objective is to show the types and representation of relational and non-relational databases, third objective is to focus about MongoDB that is a type of Document Databases under the category of NoSQL database that is a non-relational database. Illustrating an example of Hospital Management System (HMS) which is developed in PHP and uses MongoDB as the database and the last point is comparing MySQL that is a relational database with MongoDB by how to represent these two databases and how to write answers for same query in MySQL and MongoDB, then a comparison analysis by calculating the time of selection, insertion, updating and deleting between MongoDB and MySQL with the help of PHP.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

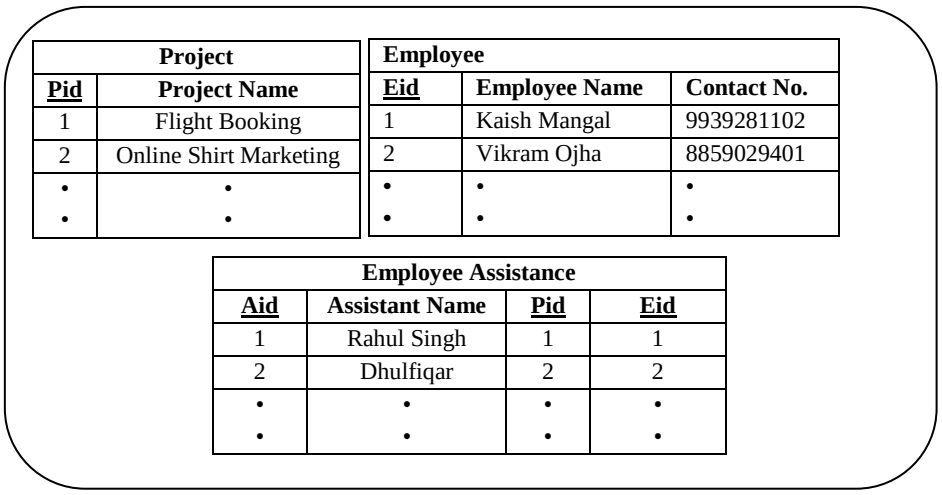
Chapter 1

Introduction

Database is an organized as collections of data, Database Management System (DBMS) is a common software to access and control this data by adding deleting and modifying data. There are different kinds of systems and models that can implement database. Traditional relational databases have been used for many years. Lately, many of NoSQL databases systems are emerging. Databases became a resource that has been used by millions of users all over the world. All the values inside the database have to be well organized so that it is going to be easier to retrieve data from database with the minimum time required. These models discussed in the following subsections.

1.1 Relational database

Relational database is most common database storage model used to store and retrieve data. It stores data in tables, every table contain rows and columns, and every table has predefined schema. This type of database works perfectly when the amount of data is small. An example of how data is stored in relational databases is shown in Figure 1.1.



Project

Pid	Project Name
1	Flight Booking
2	Online Shirt Marketing
•	•
•	•

Employee

Eid	Employee Name	Contact No.
1	Kaish Mangal	9939281102
2	Vikram Ojha	8859029401
•	•	•
•	•	•

Employee Assistance

Aid	Assistant Name	Pid	Eid
1	Rahul Singh	1	1
2	Dhulfiqar	2	2
•	•	•	•
•	•	•	•

Figure 1.1: Example of representation of relational database

As shown in Figure, there are 3 different tables, every table has different information. First table is Project contain 2 column, first column is Pid that contain the numbering

for the Project Name column, every table has to have a unique primary key that have to be one of the column in that table to make it easy for calling that record from other tables, this is how join method in relational databases work to connect more than one table in same database. So Pid column is the primary key of Project tables, second table is Employee that contain the name and contact number of each employee, the primary key of this table is the Eid that is numeric entry, the last table is Employee Assistance that contain Aid as primary key for this table, Assistance Name the name of the assistance, and Pid is the foreign to connect this table to the Project table and Eid is the foreign key to connect this table to Employee table, the joining in this database has done between tables by the last table.

For data retrieving from database in relational databases, we use SELECT processes that contain joins and projections, and for changing data in this model we use INSERT, DELETE and UPDATE processes as main commands, the Relational Database Management System (RDBMS) uses declarative language called Structured Query Language (SQL). SQL is a structured language used for retrieving, storing and modifying data in the database of relational database model, it is an ANSI standard, and there are many versions of SQL language. Relational databases have been used since long time.

Edgar Frank Codd introduced relational model in 1970 publishing a paper in communication of ACM magazine when he was working as a programmer in IBM [3]. As a research result, he gave solution to overcome data storage and access difficulty depending on rules based on relations between data. He introduced 12 rules to control data in relational model known as E.F.Codd`s 12 rules [4]. Meanwhile, System R [5], experimental database system was progressing to explain the advantages and usability of the relational model, with it was created a new language known as Sequenced Query Language (SQL) [6]. And from that time it became a standard for interacting with the database.

This type of database is facing problems, especially with programs written by object-oriented programming languages. This is because of the relational database store information data as rows and columns, and the object data model is not constrained to hold data in rows and columns. But developer made a definition known as template that completely depict a certain class of information. Every record

(object) is an example of class. Class definition also contain methods, which act onto the data described by the class. There is no similar contract in relational databases.
Relational databases also suffer from runtime issues in important scenarios. Relational databases display very poor performance characteristics for sparse tables. Sparse tables are tables where many rows have no values for many columns. Sparse table data is in semi-structured shape but relational model is a on multiple tables and costly joins processes are needed to relate them. One join identified as a "jump" along an edge between two nodes in a network.

Recently, due to cloud services like Google, Amazon data is increasing exponentially. Corporate database collect terabits of data per day. Due to social networks like Facebook and Twitter data is getting more connected and semi-structured. The relational databases are facing many new issues every day to confront these challenges. Some problems with relational databases are listed here.

a) Scalability: the scalability of the system is that its capability to cope with the growing workload [19]. There are two kinds of scaling: vertical scaling and horizontal scaling. Vertical scaling ("Scaling up") means increasing the capacity of the nodes in the system, that can be achieved by using stronger hardware (more memory, faster processor, and larger disks). In horizontal scaling ("Scaling out") more nodes are added. As RDBMS follow ACID and rich query model. The best way to fulfil this is to have dataset on a single machine that is RDBMS having vertical scaling. But an organization finds it cheaper and handy to scale out by adding more nodes rather than investing on a single node.

b) Complexity: RDBMS performance falls off as it normalizes data. As there will be more tables, table joins and thus more internal database processes for query implementation, when the database starts to increase into terabytes, the system slows down.

c) Structured data: SQL can be used handily with structured data. However, it is difficult to use this language with other kind of information like semi-structured data, because it is designed to be structured.

d) Object-relational impedance mismatch: Object-relational impedance mismatch makes is very hard and time exhaustion to fit an object oriented object into relational table.

e) Fixed schema: An issue with relational databases is schema evolution. The most powerful characteristics of relational databases that are demand of fixed data model

and referential integrity are not according to needs of changing business demand. Scaling and semi-structured data storage are the requirements of today`s businesses.
An important concept in the relational databases is transactions, Jim Gray defined there features of transaction: Atomicity, Consistency and Durability (ACD) [1]. After that Harder and Reuter added fourth one: Isolation it became (ACID) [2].

To understand more about ACID, the following is the meaning of each of them [26].

a) Atomicity mean that each operation is happened all or not happened, mean if one part of the operation fails, the operation fails, and the state of the database remain same, the atomic system have to guarantee the atomicity in all the situations, even when power failures or errors or crashes.

b) Consistency mean that system is going to ensure any operation happens in database from valid state to another valid state, all data that has been written in database have to be valid according to all guidelines.

c) Durability mean that every operation that has been finished, it remains like that, even if the operation time the power loss or errors happened. In relational databases, after a group of SQL statements carry-out, the results need to be saved temporarily to protect it against power failure, operations must be recorded in a variable memory

d) Isolation means ensuring that concurrent execution of operations gives result in the state of the system that should be gained if operations were excursed serially mean one after one. The main goal of concurrency control is to provide isolation. Depending on concurrency control, the effects on operation that did not complete might not visible for other operations.

Relational database has low scalability and issues with the controlling of huge data query time become slow [7] especially after the rise of the Web 2.0. And to overcome these problems new type of databases called as NoSQL (Not Only SQL).

1.2 Non-relational databases

Non-relational databases are the future of the database because of non-relational databases solved the problems that relational databases have, and they became more necessary after web 2.0 applications and social-network applications, that wanted millions of user reads and writes, storage of information, support and maintenance, have become the biggest challenge. Facebook and Twitter accumulate terabytes of

data every day for millions of its users [21]. Google have to save huge amount of data in the form of web pages and links. In 2008, it already passed mark of more than 1 trillion unique URLs [22]. These types of services are likely to increase massively in next few years [23].

Relational databases have increasing issues to cope with these trends [24]. Relational database management systems support SQL query language, strong uniformity and Referential Integrity. They are designed to grip few write request over a day and save data on a single instance on server. As the database increases, more tables, table joins, keys and internal database processing required for implementing quires and then the system slows down.

1.2.1 Types of non-relational databases

There are some existing non-relational databases like: Object-Oriented databases, NoSQL databases and XML databases, each one of them has different categories under them.

1.2.1.1 Object-oriented databases

Object oriented model substitute the relational model in 1985 as the controlling solution of structured data storage [20]. The object oriented model is based on the collection of the objects. The main aim of the object oriented model is bridge the gap between real world and the data model. A single language interface overcomes the resistance mismatch. This eliminates many of the inefficiencies that happen in mapping a declarative query language like SQL to an imperative language like “C”.

Unlike relational model, object oriented model is not value oriented. Object are real world objects and objects belong to class. A class include both object attributes and process definitions. In object oriented model, processes are known as methods. Methods and attributes can be defined as public or private. Private methods are apparent to only object itself and public methods are visible to other objects also. Attributes and methods can also be inherited by other classes privately or publically. Data transmission between objects can be done by passing messages to the methods. Due to difficulties of the data transmission, object oriented databases have never been in mainstream use and finally theses were subsumed under the relational databases and object-relational databases came into the existence. These databases have object-oriented front end on a relational databases. An application interfaces with this kind of

database as though the data is saved as objects. However, the system will switch the object information into data tables with rows and columns and grip the data from the same as a relational database. Similarly, when the data is called, is must be re-assembled from simple data into complex objects. As an example of this type is db4o that allow to make query with program code like java and .NET

1.2.1.2 XML databases

With the development of Internet, there was a need of standard popular data format for data exchange between applications. XML 1.0 and XML 1.1 solved this issue. XML is used in web-based and business applications to carry the data saved in the relational databases. The software systems which are used to control the XML data are known as XML databases. XML databases are non-relational databases with query languages XPath, XUpdate or XQuery. The main classes of XML databases are as follows [25].

• **XML-enabled databases (XEDs):** To save the XML data in these databases, each XML document is decomposed and its data are saved into tables. But these databases were not capable to grip documents of large sizes as decomposing and recomposing large documents meant a high number of join processes, leading to unacceptable performance, both in calling and querying them.

• **Native XML databases (NXDs):** These databases were specially developed for storing XML documents. These databases adopts XML data model for storing XML data. In these databases, hierarchy and organizing of nodes of XML documents can be preserve efficiently. This improves the performance in controlling huge XML documents.

• **Hybrid XML databases (HXD):** these databases can be treated as either a Native XML Database or as an XML Enabled Database depending on the requirements of the application.

However XML and Object-oriented databases are non-relational database but finally they are subsumed under the relational databases. XML is integrated with relational databases to enable the save, call and update of XML documents. Object-Oriented databases have subsumed under relational databases and Object relational databases came into existence. XML database is not really non-relational databases because the query of this type is XQuery XUpdate and XPath so it is not SQL, it

makes a strong data in XML format. As an example of this type is eXist that is an open source java based XML database, has same like the CouchDB a RESTful HTTP interface [14].

1.2.1.3 NoSQL databases

NoSQL database was introduced by Carlo Strozzi to refer and open source database that was not using SQL interface. Carlo Strozzi prefer to refer to NoSQL as (noseequel) or “NoRel”, which is a principal difference between that technology and already existent [8]. The start of NoSQL databases can be regarding to BigTable. Model developed by Google [9] used to store Google`s projects, like Google Earth and Google Search. Amazon developed its own dynamic database system [10].

There are four main types of NoSQL databases: Key-Value stores, Graph databases, Column stores and Document stores.

• **Key-Value Stores:** first type of NoSQL databases, they are connected arrays, having keys and values. Every key is unique and object, values could be different types like floats, integers, strings or byte arrays or hashes [12]. Figure 1.2 shows an example key-value database.

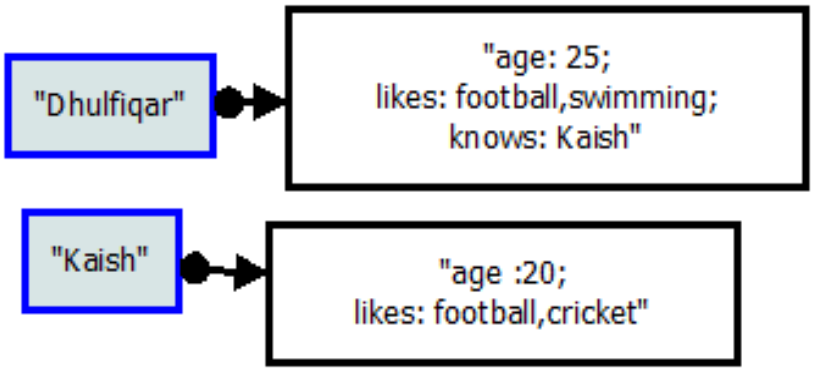


Figure 1.2: Example of key-value data store

As shown in this Figure, “Dhulfiqar” and “Kaish” the dark triangles both are the keys in this example, and keys have to be unique and keys could be numbers too, and their details on the light right triangles are the values, values could be same or different no need to be unique, as in the previous example the values vary between Dhulfiqar and Kaish, like Dhulfiqar has an information extra that he knows Kaish but Kaish does not have the knows entry. Every key connected with the value of it by a single one side arrow pointing toward its specific value.

The simple model that is given by a Key-Value store makes it work efficiently and fast. A popular example of Key-Value stores is Amazon`s Dynamo, Dynamo gives an efficient and highly scalable storage solution [13]. The main advantage of this type is that it has simple data model without any structures or relations resulting in efficient and fast management [14].VoltDB and Redis are examples of this type of database.

• **Graph databases:** Another type of database is Graph database, these databases are the best for representing data with a high number of interconnections between data. As name indicates, the data in this type is represented by graphs [15]. Figure 1.3 shows an example of Graph database.

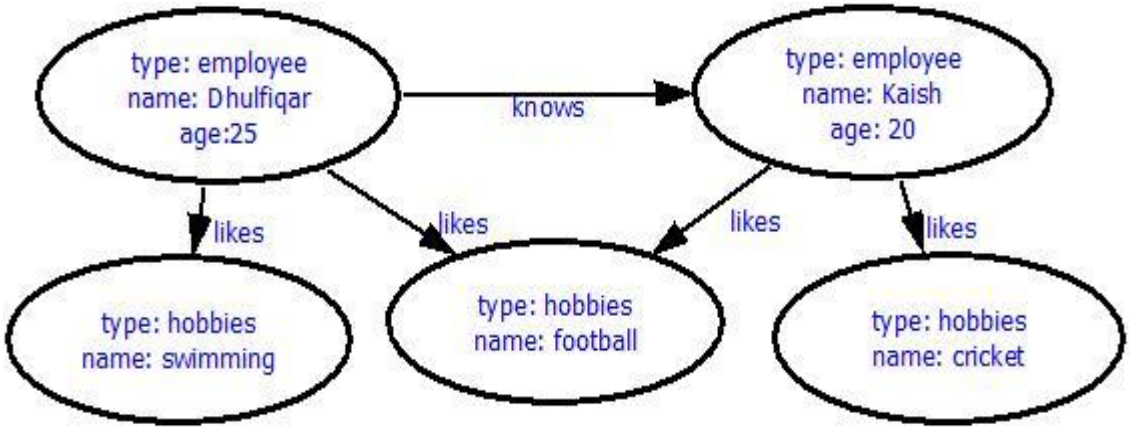


Figure 1.3: Example representation of graph database

As shown in this Figure, information stored inside ellipse is vary from ellipse to another and each ellipse could be related to others or not, like Kaish has 2 hobbies cricket and football but not swimming. For represent the relationships between ellipses are by arrow one side direction, the name of the relation is written on every specific arrow.

The main advantage of this type of database is that it has a relations same as RDBMS, but RDBMS cannot change their structure easily. And the network data are complex to manage in relational model [14]. Neo4j and OrientDB are examples of this type of databases.

• **Column Stores**: Also called Extensible Record Stores [16], the column stores and relational databases have some of their concepts same, like columns and rows. But column databases do not need for columns to be defined, a column is a key value pair, the key is the identifier and the value is where data stored, and each column can be

exist for each row, some rows or even for only one row. For the row, it can have all columns or only one, for an example of how data represented in columns store is Figure 1.4.

Key	Employee	Project	Assistant
1	Name:Dhulfiqar Age:25	Flight Booking	Rahul Singh
2	Name:Kaish Age:20	Online Shirt Marketing	Vikram Ojha

Figure 1.4: Example representation of column store

This Figure is showing an example of how data stored inside column databases, these databases have the ability of storing much more than shown, in other word, each column can store more rows and columns, thinking of that as a tables stored inside table. In relational databases we can use multiple tables to represent that and connect them by using the second table and third table primary keys and place them in the first table as foreign keys to link them together (join them), so same example if we want to present it in relational database, it is going to have 3 tables linked together employee is separated table, project separated table and assistant separated table, and each of these tables has different primary key. Column stores is easier to insert more rows and columns inside it, all these giving a result that column stores databases are faster in querying data than relational databases.

One of the recognized example of column data stores came from Hadoop world, called Apache HBase, it was designed to take control of big data, millions of millions of records and columns, if we want to make a database for all car`s from the last 100 years ago, this is the write solution for such big data amount.

• **Document oriented databases:** these type of database stores data as documents, the term documents is for the structure of data format. The main advantage of this type of database is that the schema is not fixed, different document can have different fields [14]. Each document can consist of strings, integers, arrays and document. Examples are CouchDB and MongoDB. Document oriented database use JSON format to store data.

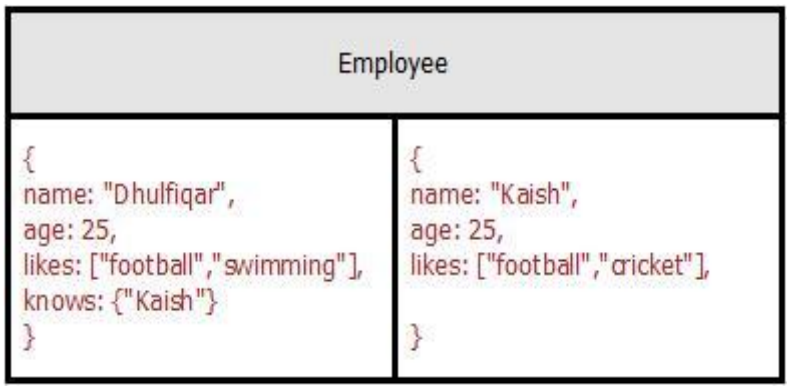


Figure 1.5: Example representation of document oriented database

In this Figure, there are two documents inside the collection called Employee, the name field is a string, the age field is an integer, the likes field is an array inside the document, and the knows field is a document inside a document. Array and document inside document is not accepted in relational databases.

There are more types of databases types like Object-Oriented database and XML database. Object-Oriented database uses the same idea of objects in the programming languages.

1.3 MongoDB

It is an open source project which has been driven by the company 10gen Inc. that also gives high professional services about MongoDB, MongoDB is developed in C++ and is schema free. The main goal of designing MongoDB is to reduce the gap between high and fast scalable Key-Value databases and the feature-rich of RDBMS [16].

The insertion of a document in MongoDB in a JSON format, it is same like BSON but BSON is binary representation and JSON is JavaScript representation. Following is an example of a doctor document representation in JSON format:

```
{
      "_id" : ObjectId("5345b6450eb961f170ffe657"),
      "name" : "Dr.Vishnu ",
      "gender" : "m",
      "age" : 60,
      "specialist" : [
            " Internist' , `Pulmonologist "
      ],
      "contact" : {
            "phoneNumber" : 7122183244,
            "email" : " vishnu@gmail.com"
      },
      "roomNo" : ObjectId("535008496239c8115d89348b")
}
```

Figure 1.6: Example of a MongoDB document

Following code, gives an example of inserting a document inside collection.

```
db.<collection name>.insert ( {"name" : "Dr.Vishnu ", "gender" : "m","age" : 60, …});
```

Figure 1.7: Inserting document inside MongoDB

In MongoDB the collection name could be generated automatically or manually. The previous line shows that how to create it automatically, for manual way follow the following code:

```
db.createCollection ( <collection name> , {<configuration parameters>} );
```

Figure 1.8: Creating a collection inside MongoDB

In this case we can identify the configuration parameters manually too, like the size of the collection, _id fields, etc [17]. MongoDB have the following different data types in same document as following:

a) Integer, Double, Float.

b) Strings.

c) Object for BSON-objects.

d) Object ID as the first line in Figure 1.6, it is automatically generated if the user do not want to enter the id manually. It is given automatically by MongoDB, and it is unique value for every document, it contain 12 byte of binary composed of the following components:

- First 4 bytes are for timestamp in seconds from the date of UNIX epoch.

- Next 3 bytes are the machine id.

- Next 2 bytes are the MongoDB process id.

- Last 3 bytes are incrementing counter.

e) Array.

f) Date (automatically or manually).

g) Document (document inside document).

Chapter 2

Problem Statement and Motivation

Millions of users are using Internet for reading and writing at the same time and for that high traffic high performance in term of scalability and secure handling of data is required. Relational database could not give all these properties, but for non-relational databases they support such options. For example, relational databases are not best fit for hierarchical data storage whereas NoSQL databases are more efficient since they store data in JSON format and very good with the big data and now a days so many organization are changing their databases from relational to no relational databases like Facebook and Twitter, a high number of users and developers start using NoSQL databases. NoSQL databases is divided into 4 categories (Key-values Stores, Column Family Stores, Document and Graph) and each category is better in handling specific use-cases. For example, database containing relationships will be best store in graph databases.

One of the categories is document oriented databases which are appropriate for web applications, which can store data and handle dynamic queries, MongoDB is a popular database under document oriented databases and provides horizontal scalability, high performance, flexibility and at the same time is more secure. MongoDB has applications in management systems, mobiles and gaming.

There are already applications that are using MongoDB. But still, by default relational databases such as MySQL, Oracle and PostgreSQL are used for development.

In this thesis, we showcase of:

- Using MongoDB as a database of a Hospital Management System using PHP.
- Benefits of using non-relational databases over relational databases by comparing between their terminologies and querying.
- Performance analysis of MySQL and MongoDB in terms of query execution time.

Chapter 3

Motivating Example

Hospital Management System (HMS) is an information system that has been designed for managing all the portions of a hospital. Such as medical treatment, reception, employees. Technical engineers are responsible for maintain hospital systems, they handle all the troubleshooting that may require to keep the system online for data patients to doctors and nurses.

In this chapter, example schema is explained in detail about Hospital Management System (HMS) in both MySQL as relational database and MongoDB as non-relational database.

3.1 ER Model

The Entity-Relationship (ER) model is data model to describe the information about a process requirements, it is used mainly for describing databases. The ER relationship model made of components that connected by lines to show the relation between these components. ER model representation for HMS database is shown in Figure 3.1.

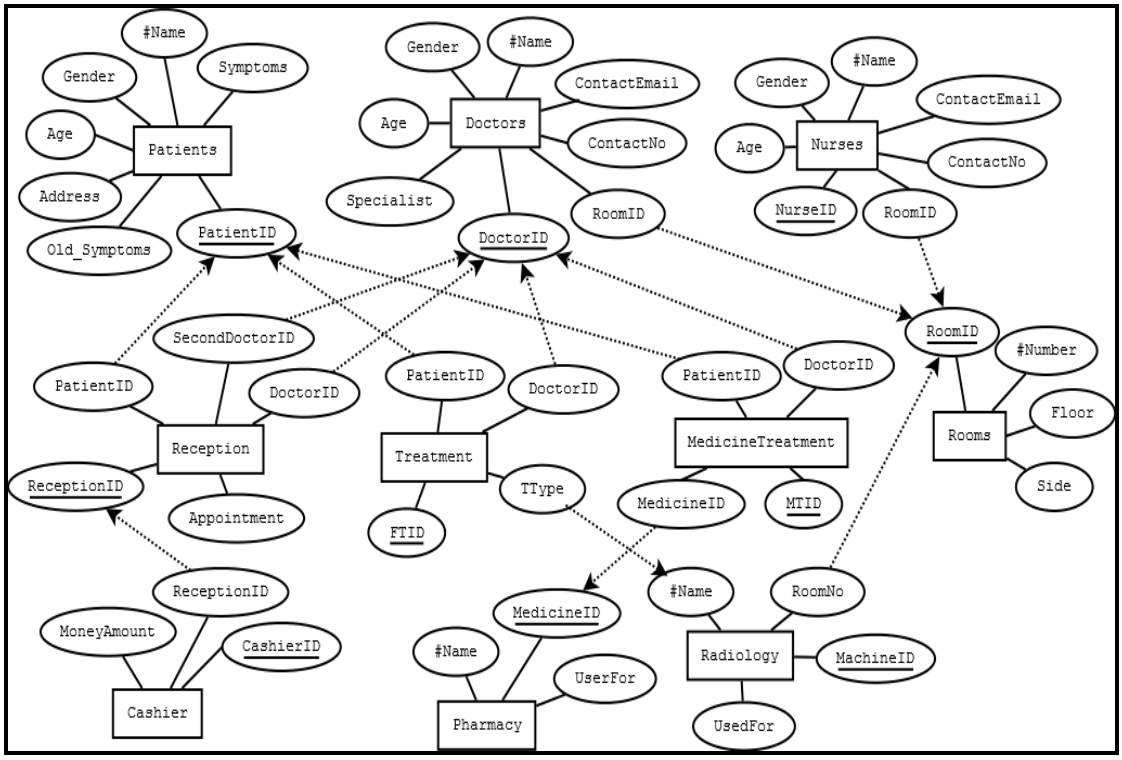


Figure 3.1: ER Diagram of HMS

As shown in this Figure HMS contain 10 different collections Patients, Doctors, Nurses, Reception, Cashier, Treatment, MedicineTreatment, Rooms, Radiology, and Pharmacy. These collections are represented as rectangular in this Figure, the ellipses are the fields in each collection, and pointing arrows describe the relation between document to document inside another collection. Each collection has one primary key. The primary key for a collection is represented by underline of the field name.

These collections inside the database are going to be explained. Reception collections contains Patient_ID that contain the id of a specific patient, Doctor_ID that contain the id of a specific doctor that going to meet the specified patient, thenWithDoctor has another doctor id, some patients going to meet more than one doctor like a patient has a broken bone first he has to meet the radiologist then the other doctor, appointmentDate has the date of appointment for the specified patient. Patients, Doctors, Nurses contain the information about each them. Treatment Table contain the patient and doctor with the type of treatment in the radiology table like MRI. MedicineTreatment contain the patient with the doctor and the medicine that is taken from pharmacy.

3.2 Class Diagram

Class diagram is a Unified Modelling Language (UML) that used to describe a structure of a system, by showing the contents of that system, attributes, and relationships between objects. Class diagram can be used for database description.

Every class diagram contains 3 parts. The upper one is the name of the class, it has to be bold and centred, the middle one is the attribute part that gives the attributed for that class, it has to be left aligned, and the last part is the methods or operations description. The last part is not needed for database description.

MongoDB uses RAM to keep the recently used data. So whenever indexes are generated to queries, all data collections fits in RAM and queries are run from it. Query cache not there in MongoDB. All queries run from indexes directly or from data files. Class diagram could be used for representing the data in MongoDB as shown in Figure 3.2.

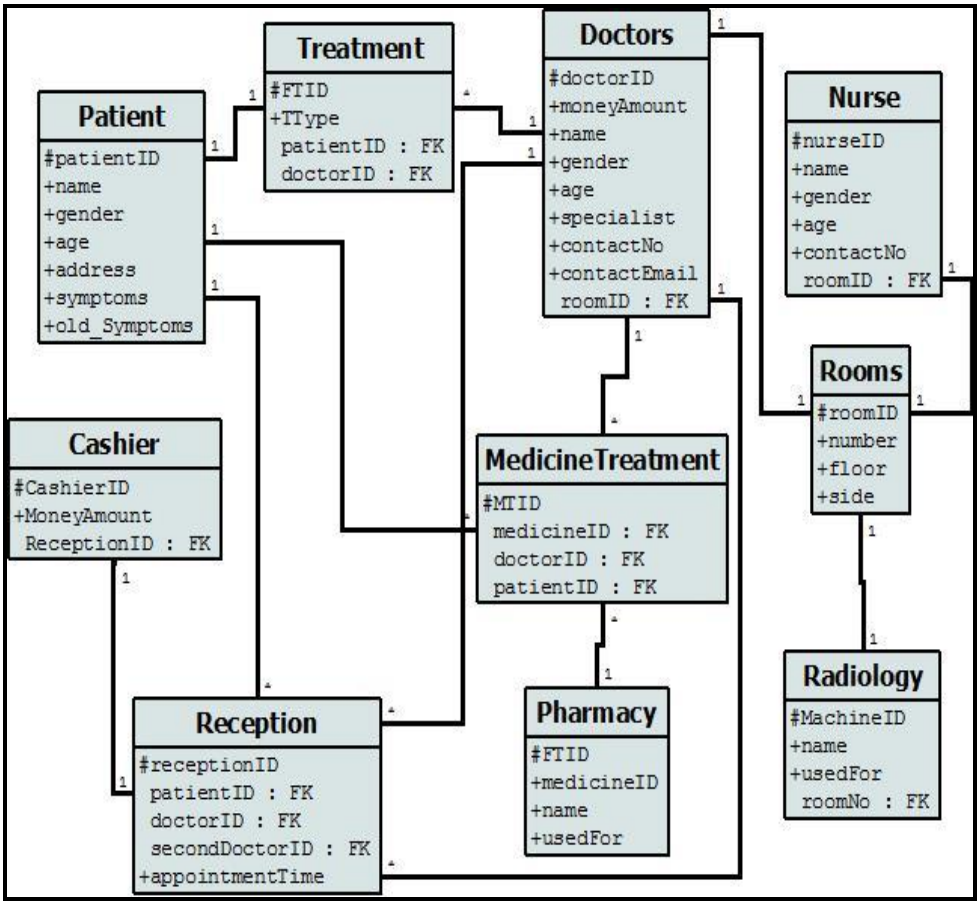


Figure 3.2: Class Diagram of HMS in MongoDB

Collections are represented by classes, the name of the class is in the upper part of each class, fields names are in second part of the class, and referencing between collections are represented by relationships that connect the classes.

Every collection has primary key field, and this field is represented by (#) before that field name, the fields without any sign before them but having FK after them, they contain the foreign keys from other collections, other fields with (+) sign are the normal fields that contain normal documents of that collection.

For the connections between collections are represented by lines, every line has two values one next to the start of it and the second one next to the end of it. If (1) on both sides of the line between two collections, that mean for each document in the first collection an only one document on the second collection, but if (*) on both sides of the line between two collections, that mean one or more document from the first collections could have one or more from the second collection, and if on side (1) and the other side (*) on the lines, that is mean one document from the first collection can has multiple documents from the second collection.

Chapter 4

MongoDB Vs MySQL

Non-relational databases have many advantages compared to relational databases, because the scalability in non-relational databases allow to be horizontal scale so the applications can scale to new levels.

Non-relational databases built for cloud like Amazon, Google can easily distributed, and there is no need for complex SQL queries.

Few advantages of non-relational databases over relational databases are as follows:

a) Flexible scaling: non-relational databases support horizontal scaling unlike the relational databases, which is going to make the increasing of server's capacity simpler.

b) Flexible data model: non-relational databases are schema less, so there are no restrictions for data modeling, that make records have different numbers of fields that make can be different from record to record.

c) Big data: because of social networks and cloud services, data in increasing very fast, that relational databases management systems gave issues to handle big data.

d) Open source: non-relational databases are open source so that they can be developed. Open source are more secure, reliable and fast.

e) No database administrator required: because off non-relational databases having the ability of automatic repair, simple data model and distribution data, so non-relational databases need less administration.

f) **Unstructured data:** relational database is suitable for structured data but for semi-structured data it is very difficult because it is designed to be structured only, whereas MongoDB is suitable for both structured and semi-structured data. Schema: The problem with the SQL is that it has fixed schema, while MongoDB schema less, so for business requirements SQL is not supporting the changing in the requirements.

g) Objects: The object relational resistance conflict make it time consuming and difficult to make and object oriented object inside a table in relational databases.

This is for the features of MongoDB over SQL, there are more differences like in querying and terms we are going to clarify it below.

4.1 MongoDB Vs MySQL: Terminology

Some terms are same in MySQL and MongoDB but the most of them are different as shown in Table 4.1.

Table 4.1: MongoDB Vs MySQL in terminology

MySQL	**MongoDB**
Execute mysql .. mysql	Execute mongod .. mongo
Database	Database
Table	Collection
Index	Index
Row	Document
Column	Field
Joining	Embedding And Linking

As table is showing that Database is same in MySQL and MongoDB but Table in MySQL database called collection inside MongoDB because the table has two dimensions, rows and columns mean fixed schema, but for collection there is no fixed schema, it is schema-less. Row in MySQL database is a record in the table, and document is a record inside the collection, Columns in MySQL database are fixed for all the records in the table, while fields are not, some documents do not have some fields others has extra fields. MySQL support joining between tables, while MongoDB support referencing and embedding between documents

4.2 MongoDB Vs MySQL: Query syntax

In this part, difference between MongoDB and MySQL in querying for select, update, delete and other operations inside patients table in HMS is shown in Table 4.2.

Table 4.2: MongoDB Vs MySQL in query syntax

SQL	MongoDB
create table patients (PatientID int, Name varchar(255),Gender varchar(255),Age int,Address varchar(255),Symptoms varchar(255),Old_Symptoms varchar(255));	db.createCollection("patients")
insert into patients values (1 , 'Kaish Mangal','M',24,' india,delhi,badli,h.no.420','Headache', Null)	db.patients.({name: "Kaish Mangal",gender:'M',age:24,address: "india,delhi,badli,h.no.420",symptoms: "Headache ",oldSymptoms, ""})
select * from patients	db.patients.find()
select Name Old_Symptoms from patients	db.patients.find({},{name:1,symptoms:1, _id:0})
select * from patients where age >= 30	db.patients.find({age:{$gte:30}})
select * from patients where age > 30 and age < 40	db.patients.find({age:{gt:30,$lt:40})
select * from patients order by age asc	db.patients.find().sort({age:1})
select * from patients where age = 26 order by name desc	db.patients.find({age:26}).sort({name:-1})
select * from patients limit 8 skip 4	db.patients.find().skip(4).limit(8)
select * from patients limit 1	db.patients.findOne()
select count (*) from patients	db.patients.count()
update patients set age = age+1 where Name= 'Kaish Mangal'	db.patients.update({name: "Kaish Mangal"},{$inc:{age:1}},{multi:true})
update patients set age = 40 where name ='Kaish Mangal'	db.patients.update({name: "Kaish Mangal"}, {$set:{age: 40}})
select patients where Name='Kaish Mangal'	db.patients.remove({name: "Kaish Mangal"})
explain select * from patients	db.patients.find().explain()

Chapter 5

Hospital Management System

In this chapter we are going to show some snapshots from the website that has been designed with the help of MongoDB as a database and PHP as a web design and development language.

First snapshot is a simple design for a home page of the HMS

Figure 5.1: Home page of the HMS.

As the first page for the hospital website it is showing special services as health assessments for free and some of health tips, for admin entry is the user name and the password on the left side of the page. And we can see there are 4 different pages linked together home, tour, about and contact. For tour it contain some sliding pictures move for the hospital from inside, the about page, it contain some information about this hospital and contact page contains number and email of the hospital as shown in Figure 5.2.

Figure 5.2: Contact page of the hospital website

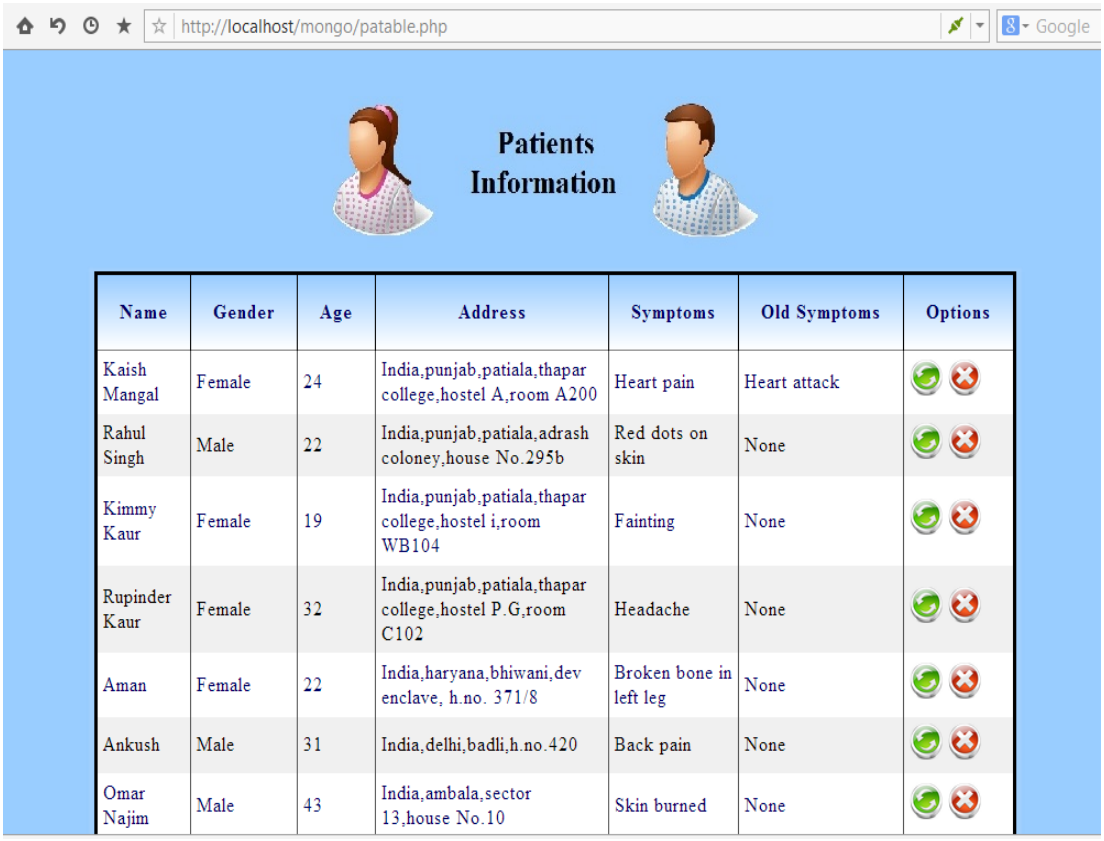
And a form that make it easier to send a direct message to the reception on the left bottom of the page that only required name, subject and the message without any registration.

After signed in as admin the home page changed and information button added to the main buttons, as shown in Figure 5.3.

Figure 5.3: Admin page

All the hospital information inside the information button. Information button contain 10 different collections patients info, doctor info, nurses info, radiology info, rooms info, pharmacy info, description treat, radiology treat, reception info and cashier info, all of them are connected to MongoDB, with this buttons, admin easily control the data inside the webpage without using the shell of MongoDB.

An example of these 10 buttons patients info as in Figure 5.4.



Name	Gender	Age	Address	Symptoms	Old Symptoms	Options
Kaish Mangal	Female	24	India,punjab,patiala,thapar college,hostel A,room A200	Heart pain	Heart attack	
Rahul Singh	Male	22	India,punjab,patiala,adrash coloney,house No.295b	Red dots on skin	None	
Kimmy Kaur	Female	19	India,punjab,patiala,thapar college,hostel i,room WB104	Fainting	None	
Rupinder Kaur	Female	32	India,punjab,patiala,thapar college,hostel P.G,room C102	Headache	None	
Aman	Female	22	India,haryana,bhiwani,dev enclave, h.no. 371/8	Broken bone in left leg	None	
Ankush	Male	31	India,delhi,badli,h.no.420	Back pain	None	
Omar Najim	Male	43	India,ambala,sector 13,house No.10	Skin burned	None	

Figure 5.4: Patients information page

This page shows all information that is required for registering every patient from name, age, gender, address, symptoms and old symptoms, the last column contains two different buttons, the green button is for edit and update the information of the specific patient in the hospital, the red button is to delete specific patient from the hospital, in the end of the page there is another button that is for inserting a new patient in the hospital like Figure 5.5.

Mariam	Female	31	India,chandighar,sector 18,house No.13	Tear in right shoulder cartilage	None	
Sparsh	Male	59	India,ambala,sector 14,house No.38	Acne	None	
Vipin	Male	31	India,haryana,bhiwani,bharat nagar,house NO.N-4	Pain in stomach	None	
Gurpreet	Male	41	India,punjab,patiala,bhadson road,house No.230a	Difficulty in breathing	None	
Chutrans	Male	32	India,punjab,patiala,thapar college,hostel J,room WA203	Heart attack	None	

Figure 5.5: Second part of patients information

In the center bottom of the page is the insertion button that is needed for registering a new patient in the hospital

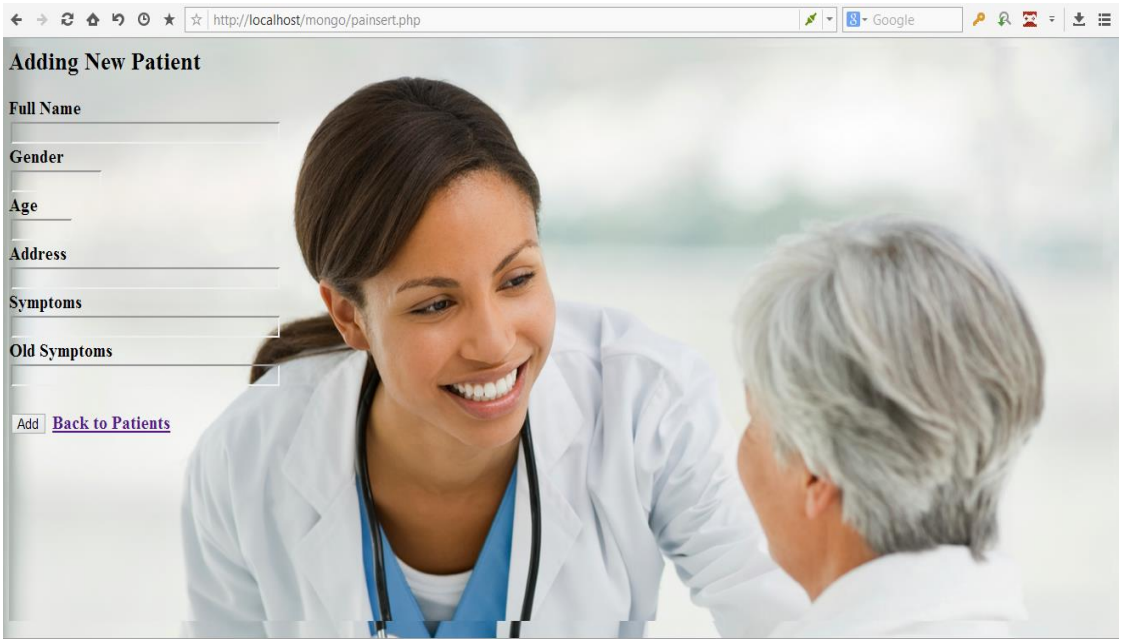


Figure 5.6: Insertion page for new patient

In this Figure is the insertion page that we can register a new patient by adding the information about him or her after finishing the inserting of patient or patients, we can go back to patient information by clicking on back to patients link next to the add button.

Doctor information page is shown in Figure 5.7.

Name	Gender	Age	Specialist	Phone Number	Email	Room Number	Options
Dr.Ali Marwan	Male	38	Cardiologist	7338462031	ali1976@yahoo.com	101	
Dr.Ahmed Hazim	Male	30	Neurological surgeon	3348802731	ahmed1984@gmail.com	102	
Dr.Dalia Hazim	Female	34	Osteopathy	9883312041	dalia1980@gmail.com	103	
Dr.Naveen Singh	Male	40	General practitioner	9833116734	naveen.singh@gmail.com	105	
Dr.Utkarish Singh	Male	29	Dermatologist	9771927410	utkarish.singh@gmail.com	104	
Dr.Vikram Ojha	Male	55	Ophthalmologist	9344719700	vikram.ojha@hotmail.com	106	
Dr.Akshama kaur	Female	31	Otolaryngologist	8122721852	akshama.kaur@gmail.com	107	
Dr.Dhulfiqar Hazim	Male	49	Radiologist	7118340834	dr.dhulfiqar@gmail.com	108	
Dr.Ehab							

Figure 5.7: Doctors information page

In this Figure we can see that all doctors in hospital are in this page, with the same last column like the patient, but this for editing the information about specific doctor or for dismiss him or her from hospital, For hiring a new doctor we need to press on the add button in the end of the page, after pressing it we are going to see page same like the Figure 5.8.

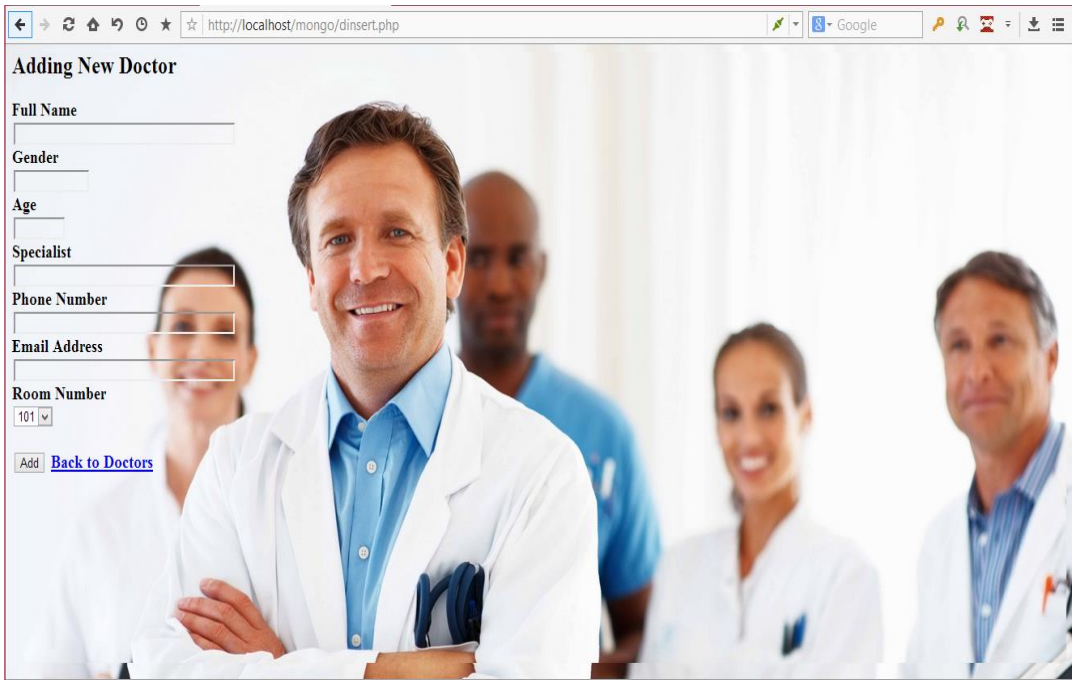


Figure 5.8: Insertion page for hiring new doctor in the hospital

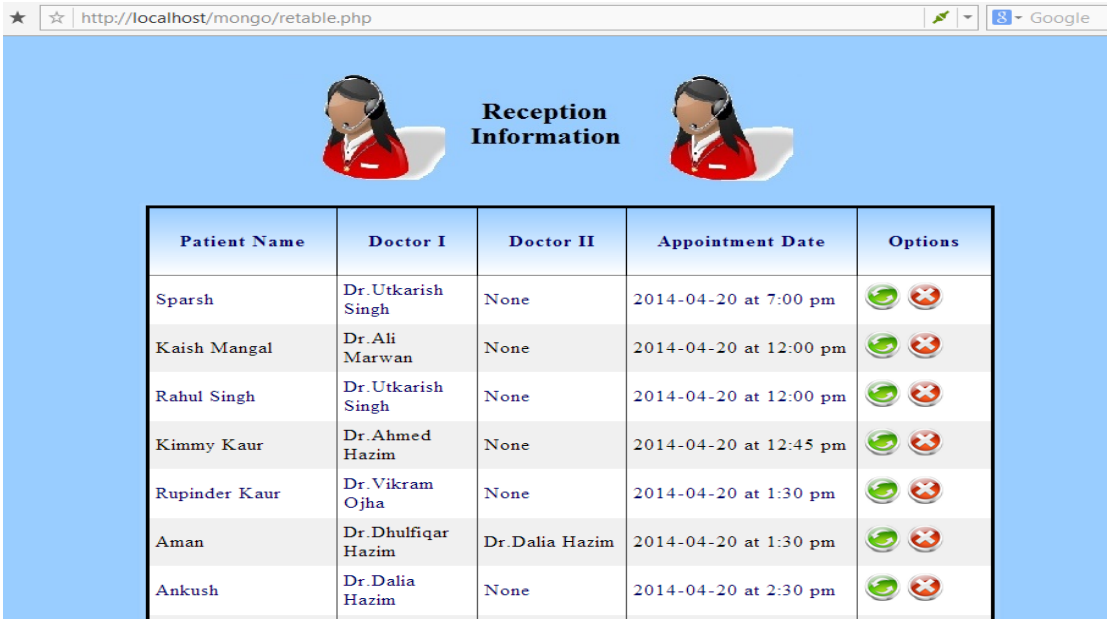
In this page we can see the fields required for new doctor registration. But the last information required to finish registration is room number that is already referenced to the room's information collection inside the database, as MongoDB is not supporting joining like SQL so referencing between collections required and the room number drop list in Figure 5.8 is containing all rooms inside the hospital.

For nurses that are required for helping doctors in hospital they have their own collection in database as in the Figure 5.9.

Figure 5.9: Nurses information page

Nurses information page containing all the information for every nurses as shown in Figure 5.9, and some of nurses do not have rooms, they are assistance when required only, the nurses collection little similar to doctors collection, both have information for each of them, and both have rooms only difference that some nurses do not have specific room. Insertion, updating and deleting are similar to doctors information page.

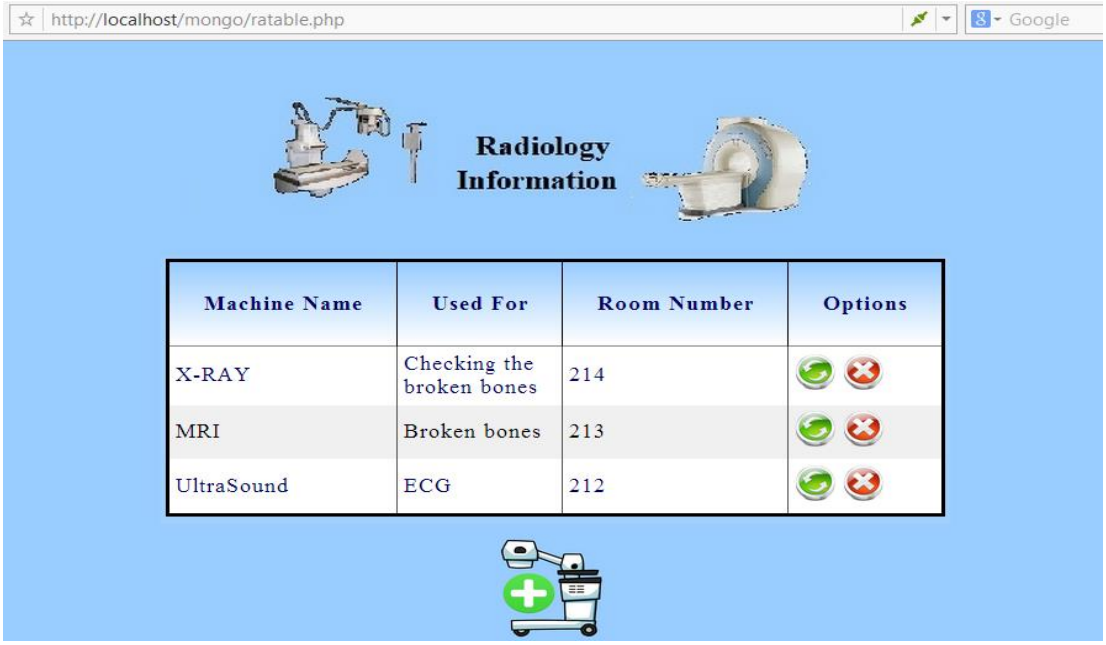
Every doctor has specific patient to treat depend on the symptoms of that patient, like patient Kaish has an appointment with doctor Ali Marwan with the date and time for the appointment, most patients have appointment with one doctor, but some patients have appointment with two doctors like Aman, she has appointment with two different doctors as in Figure 5.10.



Patient Name	Doctor I	Doctor II	Appointment Date	Options
Sparsh	Dr.Utkarish Singh	None	2014-04-20 at 7:00 pm	
Kaish Mangal	Dr.Ali Marwan	None	2014-04-20 at 12:00 pm	
Rahul Singh	Dr.Utkarish Singh	None	2014-04-20 at 12:00 pm	
Kimmy Kaur	Dr.Ahmed Hazim	None	2014-04-20 at 12:45 pm	
Rupinder Kaur	Dr.Vikram Ojha	None	2014-04-20 at 1:30 pm	
Aman	Dr.Dhulfiqar Hazim	Dr.Dalia Hazim	2014-04-20 at 1:30 pm	
Ankush	Dr.Dalia Hazim	None	2014-04-20 at 2:30 pm	

Figure 5.10: Reception information page

And that is because she has a special case that she got broken bone so that she has to see a Radiologist doctor, as we can see she has appointment with doctor Dhulfiqar before she is going to meet doctor Dalia Hazim, because doctor Dhulfiqar is a radiologist, so the first treatment is the radiology for checking the injuries with machines like X-Ray. Figure 5.11 showing the main machines for radiology section in HMS.



Machine Name	Used For	Room Number	Options
X-RAY	Checking the broken bones	214	
MRI	Broken bones	213	
UltraSound	ECG	212	

Figure 5.11: Radiology information page

As we can see in this Figure the same buttons for update, add and delete so that we can change the location of specific machine or adding another one or deleting it. So HMS has 3 machines, every machine in different room.

Doctor Dhulfiqar is the responsible for checking patients by all these machines as in Figure 5.12.

Figure 5.12: Radiology treatment information

In this Figure all patients are under doctor Dhulfiqar because he is the radiologist in the hospital. Patients have different times as we can see in Figure 5.10. After radiology session finished, every patient has to go to the second doctor for finishing the healing and knowing what medicine they have to take.

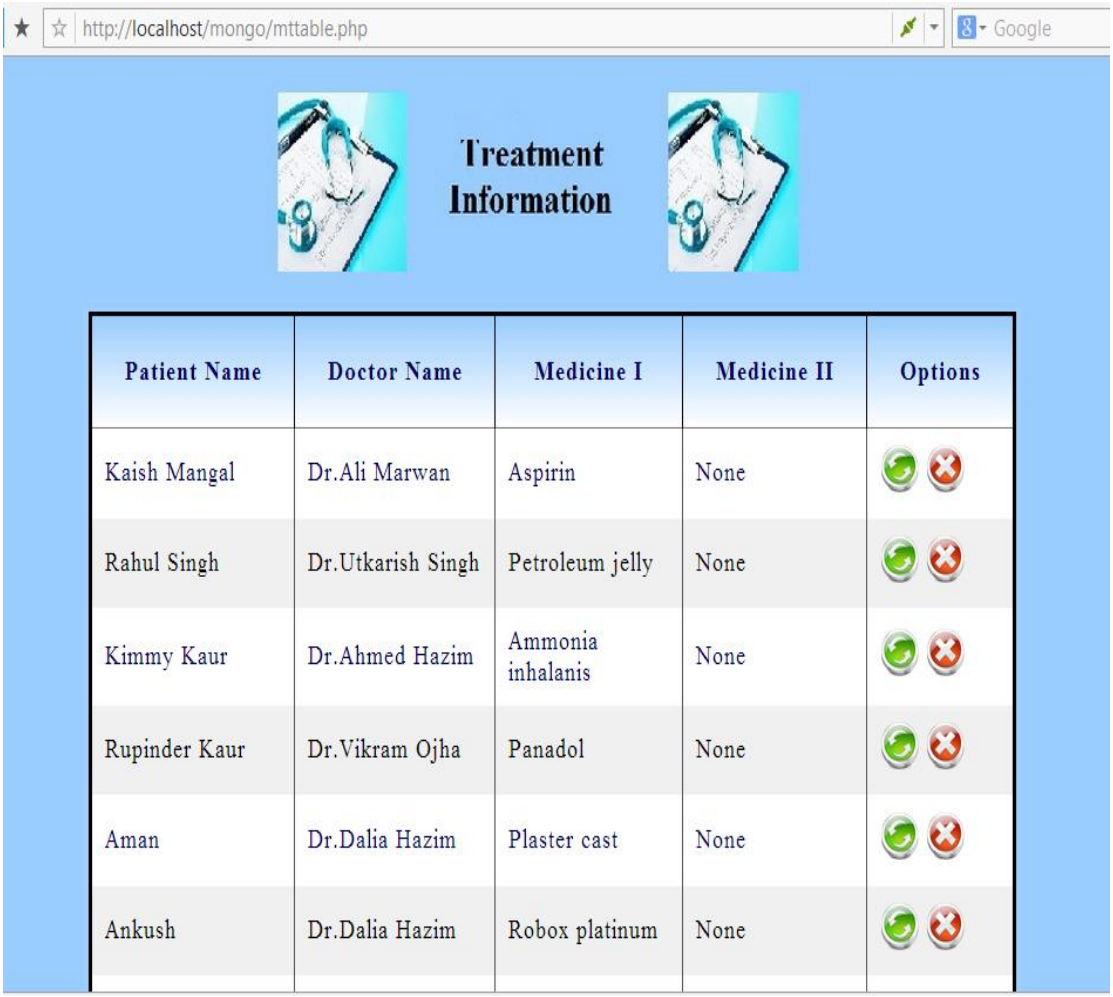


Figure 5.13: Treatment information

In this page we can see that the patient with his doctor and which medicine has been described by the specific doctor is under the Medicine I, and Medicine II if there is another medicine described from the same doctor to the same patient. There are the 3 buttons in here too for update, add or deleting the record. For make it clearer, Figure 5.14 is showing how fields in this page got inputted.

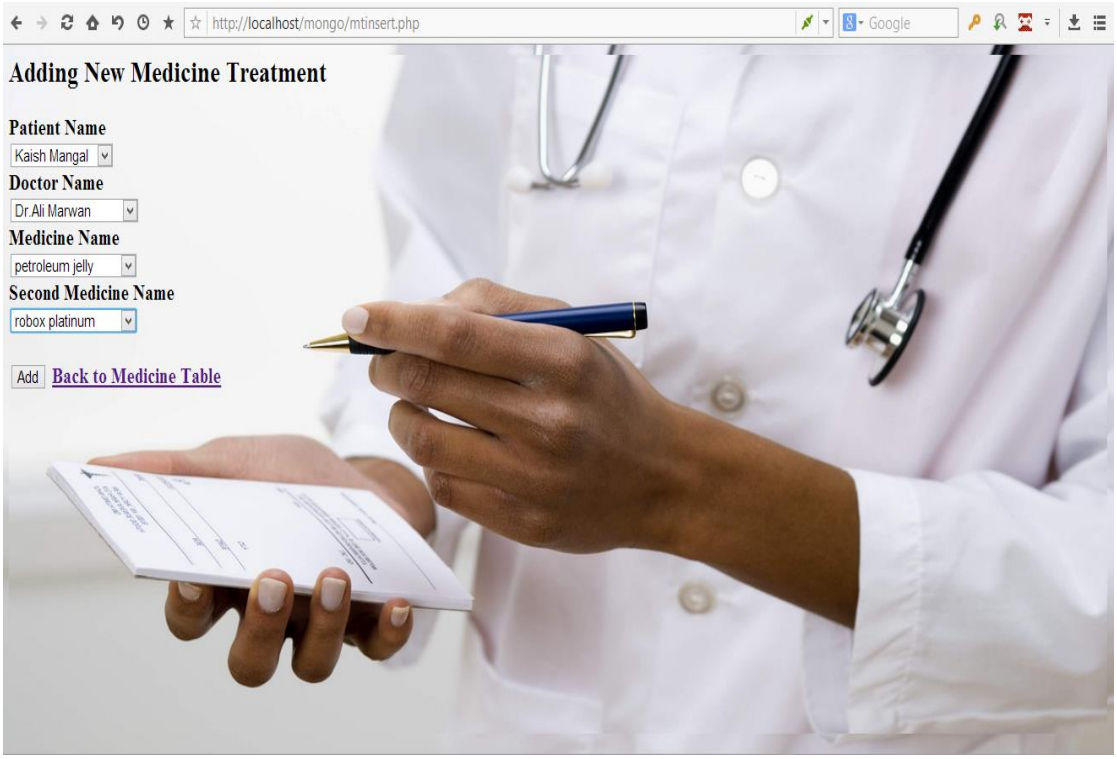


Figure 5.14: Insertion of new treatment

In this Figure patient name is a drop down list for all patients registered in this hospital retrieved from patients database collection, Doctor Name is a drop down list too but it is for doctors retrieved from doctor collection inside the database. Medicine Name and Second Medicine Name both are drop down lists for medicine retrieved from pharmacy database. After pressing add a new field is going to be added in the database and medicine treatment page.

Chapter 6

Performance Analysis

For making a comparison between relational and non-relational databases, we used MySQL and MongoDB on a machine that has a processor of Inter Core i5 2.5GHz with a dynamic memory (RAM) of 4GB. We wrote test application in PHP. We installed version 5.6 for MySQL and version is 2.5 for MongoDB. We used the four main operations select, inserting, updating and deleting for patient's information table/collection and the data that has been used in MySQL and MongoDB are same.

The first operation is selection operation as in Table 6.1.

Table 6.1: Comparison of MySQL and MongoDB for selection

Number of records / documents selected	**Time in Milliseconds**	
	MySQL	**MongoDB**
10	0.835	0.005
100	3.443	0.316
1000	439.635	2.968
10000	5300.675	35.270
100000	8055.254	910.163
1000000	131868.633	20717.802

In this table MongoDB from the first record when the records are 10 only is advanced on MySQL with high difference approximately 160 times more in MongoDB than MySQL, MongoDB in all records taking less time than MySQL, and when the records reached 1000000 MongoDB became almost 10 times more than SQL.

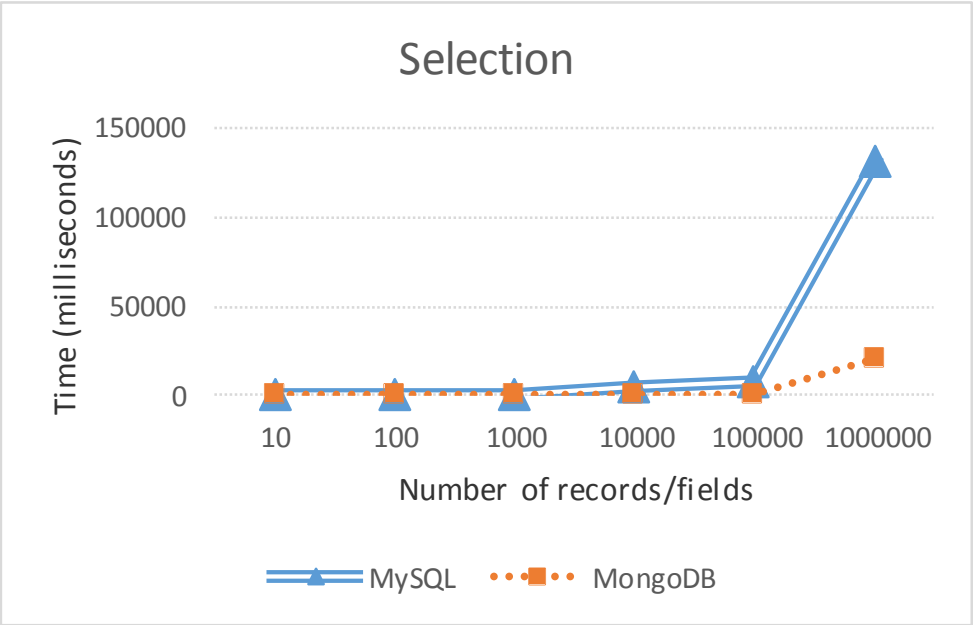


Figure 6.1: Comparison of MySQL and MongoDB for selection

This Figure is showing that how really big difference between MongoDB and MySQL, in the first two values not very clear because of the small numbers but the difference start to be clear after the 1000 records that has been selected in both databases, till the last value of selection it became very clear to see the difference between MongoDB and MySQL.

The second operation is insertion as in Table 6.2.

Table 6.2: Comparison of MySQL and MongoDB for insertion

Number of records / documents inserted	**Time in Milliseconds**	
	MySQL	**MongoDB**
10	2.425	1.452
100	33.257	14.878
1000	125.438	47.464
10000	845.064	301.711
100000	7613.270	2203.102
1000000	79161.448	25336.210

In this table it is shown that the insertion for a records (documents) in both MySQL and MongoDB is slightly difference but when the data insertion increased, the gap between them increased.

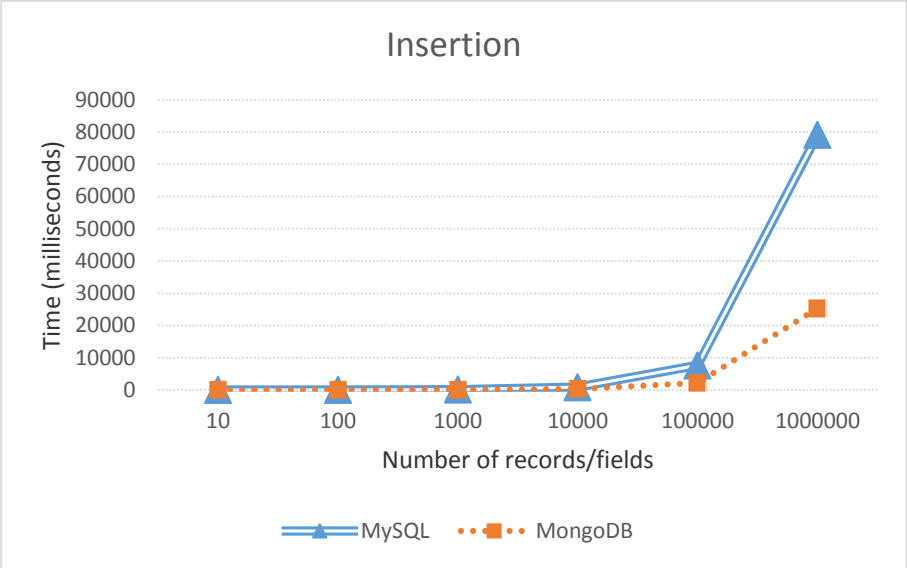


Figure 6.2: Comparison of MySQL and MongoDB for insertion

As Figure showing that in the first the difference even not very obvious but by increasing the number of insertion the difference became very clear between them.

Second operation is the updating of the data that have already been inserted in the both databases. The updating is by the name of the patient as shown in Table 6.3.

Table 6.3: Comparison of MySQL and MongoDB for updation by name

Number of records / documents updated by name	**Time in Milliseconds**	
	MySQL	**MongoDB**
10	1.058	0.319
100	25.945	2.486
1000	101.211	32.368
10000	942.967	1432.561
100000	9783.173	22616.761
1000000	97502.703	163726.819

MongoDB showing worse results than MySQL in updating by name of the patient because MongoDB updated a non-indexed column, and MongoDB is unstructured database so it forces any unindexed queries to perform complex lookups on each element in every document entry. For small amount still MongoDB is faster but with the increasing of the entire collection it take much time than MySQL.

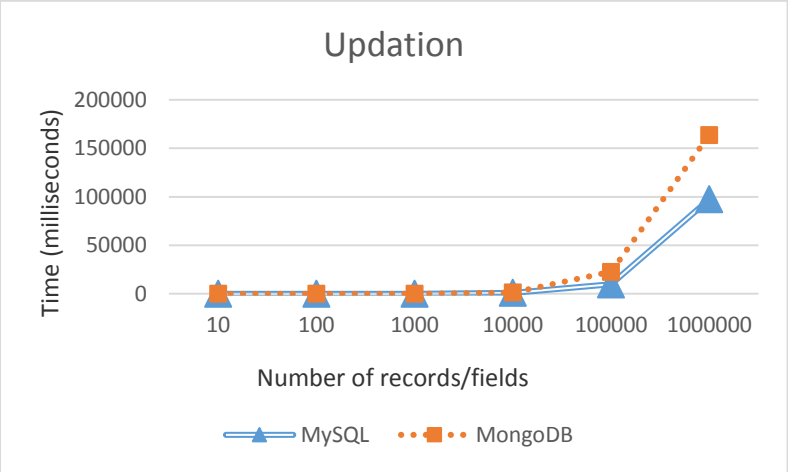


Figure 6.3: Comparison of MySQL and MongoDB for updation by name

In this Figure, we can see that MySQL is better than MongoDB when the updating was done depending on the name of the patient instead of the primary key of that document. Next is the updating of the same table depending on the id of the patient instead of the name.

Table 6.4: Comparison of MySQL and MongoDB for updation by id

Number of records / documents updated by id	**Time in Milliseconds**	
	MySQL	**MongoDB**
10	1.538	0.119
100	23.394	1.491
1000	93.163	4.193
10000	719.720	17.193
100000	7119.139	317.182
1000000	83063.196	3811.109

For updating depending on the id MongoDB shows so much better than SQL from the 10 records/documents it is better with high gap, and the gap continues to grow bigger with the increasing of the data. And for make it more clear Figure 6.5.

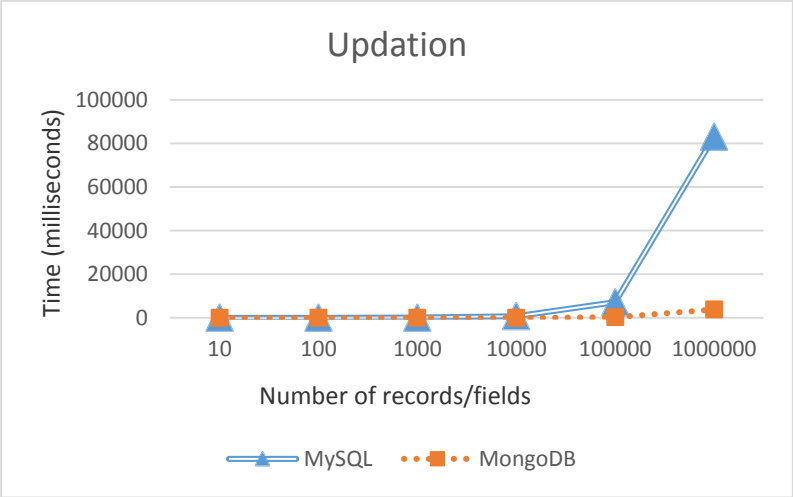


Figure 6.4: Comparison of MySQL and MongoDB for updation by id

In this Figure MongoDB is dealing so well with the updating by id because of MongoDB has a pre-built index for the primary key of a document which make it faster than MySQL primary key.

The last operation is the delete operation of the documents that we inserted before. For MongoDB it is showing outstanding performance comparing to MySQL as in Table 6.5.

Table 6.5: Comparison of MySQL and MongoDB for deletion

Number of records / documents deleted	**Time in Milliseconds**	
	MySQL	**MongoDB**
10	1.873	0.133
100	11.603	0.146
1000	102.599	53.237
10000	1079.199	411.422
100000	10265.825	1487.701
1000000	99183.286	2014.852

MongoDB handling the deleting of the documents better than all the other operations comparing to overall on MySQL. From the 10 documents deleting it is heading MySQL with large gap.

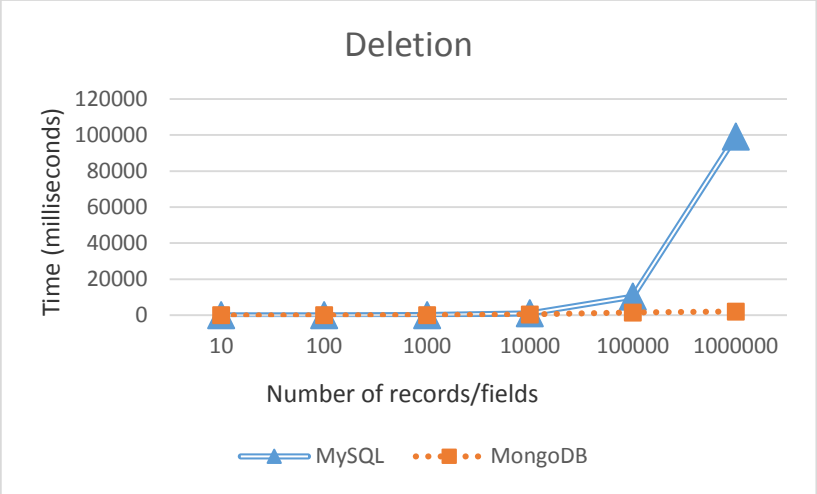


Figure 6.5: Comparison of MySQL and MongoDB for Deletion

As overall results, MongoDB is faster and better in handling data for all operations except updating by the name.

Chapter 7

Conclusion and Future Scope

7.1 Conclusion

NoSQL databases have many advantages over relational database and more and more companies are opting for NoSQL databases. Relational databases are not best fit for hierarchical data storage whereas NoSQL databases are more efficient since they store data in JSON format and very good with the big data and now a days so many organization are changing their databases from relational to no relational databases like Facebook and Twitter, a high number of users and developers start using NoSQL databases. NoSQL databases is divided into 4 categories (Key-values Stores, Column Family Stores, Document and Graph) and each category is better in handling specific use-cases. For example, database containing relationships will be best store in graph databases.

We focussed on document oriented databases in this thesis, which are appropriate for web applications, which can store data and handle dynamic queries. MongoDB is a popular database under document oriented databases and provides horizontal scalability, high performance, flexibility and at the same time is more secure. MongoDB has applications in management systems, mobiles and gaming.

In this thesis, we implemented hospital management system using MongoDB as backend and PHP for frontend. MongoDB have its own query language called MongoDB query language, whose syntax is very different from syntax of commonly used query language SQL. We compared the performance of MongoDB and MySQL in terms of query execution time. It is analysed that for less number of records, performance of both the databases are comparable. But as number of records increases, MongoDB outperforms MySQL.

7.2 Future Scope

Possible research problems that can be carried out further, can be:

- In this thesis, only a subset of functionality of hospital information system is implemented. Functionality of the implemented system can be increased to put more features.

- Data pertaining to patients, patient's symptoms can be mined and visualised using different mining and visualisation techniques.
- Use of another class of NoSQL data store, like graph database Neo4j or key value data store Redis can be incorporated.

References

[1] J. Gray. The transaction concept: virtues and limitations (invited paper). In Proceedings of the seventh international conference on Very Large Data Bases - Volume 7, VLDB ’81, pages 144–154. VLDB Endowment, 1981.

[2] T. Haerder and A. Reuter. Principles of transactionoriented database recovery. ACM Comput. Surv., 15(4):287–317, Dec. 1983.

[3] Codd. E. F. 1970. A relational model of data for large shared data banks. Communications of ACM 13, 6 (June 1970), 377-387. doi=10.1145/362384.362685.

[4] Codd. E. F. 1985. “Is your DBMS Really Relational?” and “Does your DBMS Run by the Rules?” Computer World, October 14 and October 21.

[5] M.M. Astrahan, A history and evaluation of system R, Performance Evaluation, Volume 1, Issue 1, January 1981, Page 95, ISSN 0166-5316, 10.1016/0166-5316(81)90053-5.

[6] Donald D. Chamberlin, Raymond F. Boyce: SEQUEL: A Structured English Query Language. SIGMOD Workshop, Vol. 1 1974: 249-264.

[7] Jayathilake, D.; Sooriaarachchi, C.; Gunawardena, T.; Kulasuriya, B.; Dayaratne, T., "A study into the capabilities of NoSQL databases in handling a highly heterogeneous tree," Information and Automation for Sustainability (ICIAfS), 2012 IEEE 6th International Conference on , vol., no., pp.106,111, 27-29 Sept. 2012. doi: 10.1109/ICIAFS.2012.6419890.

[8] Lith, Adam; Jakob Mattson (2010). "Investigating storage solutions for large data: A comparison of well performing and scalable data storage solutions for real time extraction and batch insertion of data". Göteborg: Department of Computer Science and Engineering, Chalmers University of Technology.

[9] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E. Gruber. 2006. Bigtable: a distributed storage system for structured data. In Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation - Volume 7 (OSDI '06), Vol. 7. USENIX Association, Berkeley, CA, USA, 15-15.

[10] DeCandia Giuseppe, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall, and Werner Vogels. 2007. Dynamo: amazon's highly available key-value store. In Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles (SOSP '07). ACM, New York, NY, USA, 205-220.

[12] S. Edlich, A. Friedland, J. Hampe, and B. Brauer. NoSQL: Einstieg in die Welt nichtrelationaler Web 2.0 Datenbanken. Hanser Fachbuchverlag, 10 2010.

[13] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels. Dynamo: amazon's highly available key-value store. SIGOPS Oper. Syst. Rev.41(6):205–220, Oct. 2007.

[14] S. Weber. NoSQL databases. University of Applied Sciences HTW Chur, Switzerland, 2010.

[15] R. Angles and C. Gutierrez. Survey of graph database models. ACM Comput. Surv.40(1):1:1–1:39, Feb. 2008.

[16] 10gen, Inc: mongoDB. 2010. –http://www.mongodb.orgv

[17] Chodorow, Kristina ; Banker, Kyle ; Hernandez, Scott: mongoDB Manual – Collections. June 2010. Wiki article, version 5 of 2010-06-07. http://www.mongodb.org /display/DOCS/Collections

[18] http://blog.michaelckennedy.net/2010/04/29/mongodb-vs-sql-server-2008-performance-showdown/

[19] Bondi A. B., "Characteristics of scalability and their impact on performance," in *Proceedings of the 2nd international workshop on Software and performance*, New York, NY, USA, 2000, pp. 195–203.

[20] Atwood T., "An Object-Oriented DBMS for Design Support Applications," in *Proceedings of the IEEE COMPINT 85*, 1985, pp. 299-307.

[21] Vardanyan M. (2011, May) Picking the Right NoSQL Database Tool. [Online]. http://blog.monitis.com/index.php/2011/05/22/picking-the-right-nosql-database-tool/?attest=true&opt=vers

[22] Burkhart H., Rizzotti S. and Ruflin N., "Social-Data Storage Systems," in *Databases and Social Networks, Athens*, Greece, June, 2011.

[23] Higginbotham S. (2011, Sep) Sensor networks top social networks for big data. [Online]. http://gigaom.com/2010/09/13/sensor-networks-top-social-networks-for-big-data-2/

[24] Neubauer P. (2010, May) Graph Databases, NoSQL and Neo4j. [Online]. http://www.infoq.com/articles/graph-nosql-neo4j

[25] Zamboulis L., Poulovassilis A. and Papamarkos G., "XML Databases," Computer Science Information System, Birkbeck College, Uni. London,.

[26] http://en.wikipedia.org/wiki/ACID

List of Research Publications

- Dhulfiqar Hazim, Karamjit Kaur, “Implementing Health-care informatics using MongoDB”, communicated to International Journal of Database Management Systems (IJDMS).
- Dhulfiqar Hazim, Karamjit Kaur, “Performance analysis of MongoDB and MySQL for health-care information systems”, submitted in International conference on Computational Science, Engineering and Information Technology (CCSEIT) to be held at AIT Pune in August 2014.