MigDB

Relational to NoSQL Mapper

Project ID: 16-015

Project SRS Document

B.Sc. Special (Honors) Degree in Information Technology

Submitted on 01/04/2016

IT13075422 Padmasiri H.R.K.L

Team Members

Student ID	Name	Signature
IT13001476	Liyanaarachchi L.A.G.C	
IT13088156	Madhusanka K.P.L.K	
IT13093938	Kothalawala K.R.M.N	
IT13075422	Padmasiri H.R.K.L	

Supervisor

……

Dr. Anuradha Karunasena

Co-Supervisor

……

<Name of the Co-supervisor>

Declaration

I declare that this is my own work and this Software Requirements Specification does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

…….

Padmasiri H.R.K.L

IT13075422

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose

The purpose of the Software Requirement Specification (SRS) document is to provide a detailed overview of the user, system, functional and non-functional requirements of the project 'MigDB' which converts relational database into Mongo database with table structures, relationships and data in a relational database. Intended audience for the document is research teams, developers and database administrators. This document organize the system core functionalities in a well ordered manner and it helps developers in gaining clear understanding about functionalities of the system. It also helps in ensuring that the product after development meets the exact functionalities it was intended for.

1.2 Scope

The system 'MigDB' is a standalone database migration tool. The primary objective of this system is to convert MySQL database into Mongo database with SQL table structures, data and relationships without requiring JSON based or MongoDB command knowledge from the user. To achieve the main objective of the system, it is divided into three main components. Under the main components there are many sub components as mentioned below

- Relational Database Handling Module
  1. **Connection Manager**
  2. **SQL DB Analyzer**
  3. SQL DB Modification Evaluator
- Mapping Module
  4. **One-to-one mapping**
  5. One-to-many mapping
  6. Many-to-many mapping
  7. Semantic Network for Collection Mapping
- MongoDB Management Module
  8. MongoDB Collection Modifier UI
  9. MongoDB Modifier Evaluator
  10. Conversion of Referencing to embedding
  11. Conversion of Embedding to Referencing
  12. MongoDB Data Manipulator

13. Graphical Query Generator

14. SQL Query Converter

This document covers the highlighted sub components Connection Manager, SQL DB Analyzer, One-to-one Mapping and Graphical Query Generator in detail.

Connection manager has the capability to establish connections with MySQL databases and Mongo databases. This module has capability to handle multiple database connections with a local machine or remote server. The Connection Manager module helps users in easily selecting an existing MySQL database for migration with use of its MySQL database connections and it also facilitates users to manage and manipulate MongoDB databases by using its connection to MongoDB server.

The SQL DB Analyzer module is responsible for analyzing the structure of the MySQL database to be migrated accompanied by the creation of the intermediate JSON file. If the user desires to provide the database to migrated as a SQL file instead of selecting a database synchronized from the MySQL connection, this module allows user to upload a SQL file or a dump file of a database. After the upload, the module examines the uploaded SQL file for any syntax errors. In case of any syntax errors, the SQL DB Analyzer module notifies the user of the error and otherwise it analyses the table structures and creates the intermediate JSON file. The JSON file contains objects for both tables and data while including all related details and is used by all modules until the migration process completes.

One-to-one Mapping module concentrates on mapping one-to-one relationships of the MySQL database to be migrated into MongoDB. The module uses embedding to map one-to-one relationships within the database to be migrated. The one-to-one mapping module is also responsible of creating the JSON file which contains MongoDB document objects which is later to be inserted into collections.

Graphical Query Generator module facilities users to generate MongoDB statements using the graphical user interface provided. This module is especially beneficial to the novice users who are less familiar with JSON or MongoDB commands as the module is able to generate MongoDB commands by having user only clicking and typing values. The Graphical Query Generator module is capable of taking the connected MongoDB databases as the input and graphically listing the databases and collections which currently exist for users to query on. The modules supports users in generating MongoDB command for creating/ dropping

databases, creating/ modifying collections, insert/ update/ delete documents as well as querying and projection of documents without requiring any JSON or MongoDB commands.

1.3 Definitions, Acronyms and Abbreviations

JSON	JavaScript Object Notation
DB	Database
UI	User Interface
SQL	Structured Query Language
JDK	Java Development Kit
IDE	Integrated Development Environment

1.4 Overview

The end result of 'MigDB' is to convert MySQL database into MongoDB along with existing data, table structure, relationships and existing database queries consuming incremental machine knowledge and human knowledge without requiring JSON or MongoDB command knowledge from the user. The target audience of 'MigDB' is developers and database administrators who wish to migrate their existing MySQL databases into MongoDB. Sub objectives mentioned below works together to achieve the main objective of the system.

1. **Connection Management module to establish connection with MySQL database and MongoDB database in local machine or remote server while having the capability to hold multiple database connections.**
2. **SQL DB Analyzer module to facilitate upload of an SQL file and to analyze dump file and understand table structures, relationships along with data and then to map these information into intermediate JSON file.**
3. SQL DB Modification Evaluator module to graphically represent the MySQL database structure, allow user to make changes, evaluate changes for any violations and then record the user changes to the database structure.
4. Neural network as a separate module to predict about database mapping technique with considering references, column count and data types of columns. This module has capability to trigger scheduled Neural Network supervised learning

sessions using a CSV file. This module has capability to retrieve information from Neural

Network and send to other modules through a restful interface.

5. **One-to-one mapping module to map one-to-one relationships in MySQL database into MongoDB with data migration.**
6. One-to-many mapping module to map one-to-many relationships in MySQL database into MongoDB by using embedding or referencing.
7. Many-to-many mapping module to map many-to-many relationships in MySQL database into MongoDB by using embedding or referencing.
8. MongoDB Collection Modifier UI module to illustrate collection structure generated by the system.
9. MongoDB Modification Evaluator module to enable user to change collection structure by applying user experience on database mapping in user friendly way and to evaluate user changes on mapping.
10. Conversion of Referencing to Embedding module to convert existing reference mapping into embedded mapping and update database.
11. Conversion of Embedding to Referencing module to convert existing embedded mapping into reference mapping.
12. MongoDB Data Manipulator module to access NOSQL database on localhost and graphically represent data to the user and enable user to manipulate data and save changes.
13. **Graphical Query Generator module to generate MongoDB commands with the use of interactive graphical user interface.**
14. SQL Query Converter module to convert given MySQL statements into MongoDB statements.
15. User friendly web portal to enable user for accessing above services with getting continuous user interaction. This web portal has capabilities to handle payments, file upload and download, user management and file version management.

The overall descriptions, specific requirements and supporting information of the highlighted modules would be discussed in the rest of the document. The purpose of the document, the scope of the document, the benefits, objectives and goals of the system were covered in the first chapter of the document. The second chapter of the document gives a general description of the modules Connection Manager, SQL DB Analyzer, One-to-one Mapping and Graphical

Query Generator and this chapter compares the system with other system currently available and also gives an overview on system interfaces, user interfaces, hardware interfaces, software interfaces, communication interfaces and memory constraints. The second chapter is intended for non-technical users and also explains product functions, user characteristics and assumptions. The third chapter thoroughly explains interface requirements, performance requirements, design constrains, system attributes and other requirements as it is intended for technical users. The forth chapter provides the supporting information.

2 Overall Descriptions

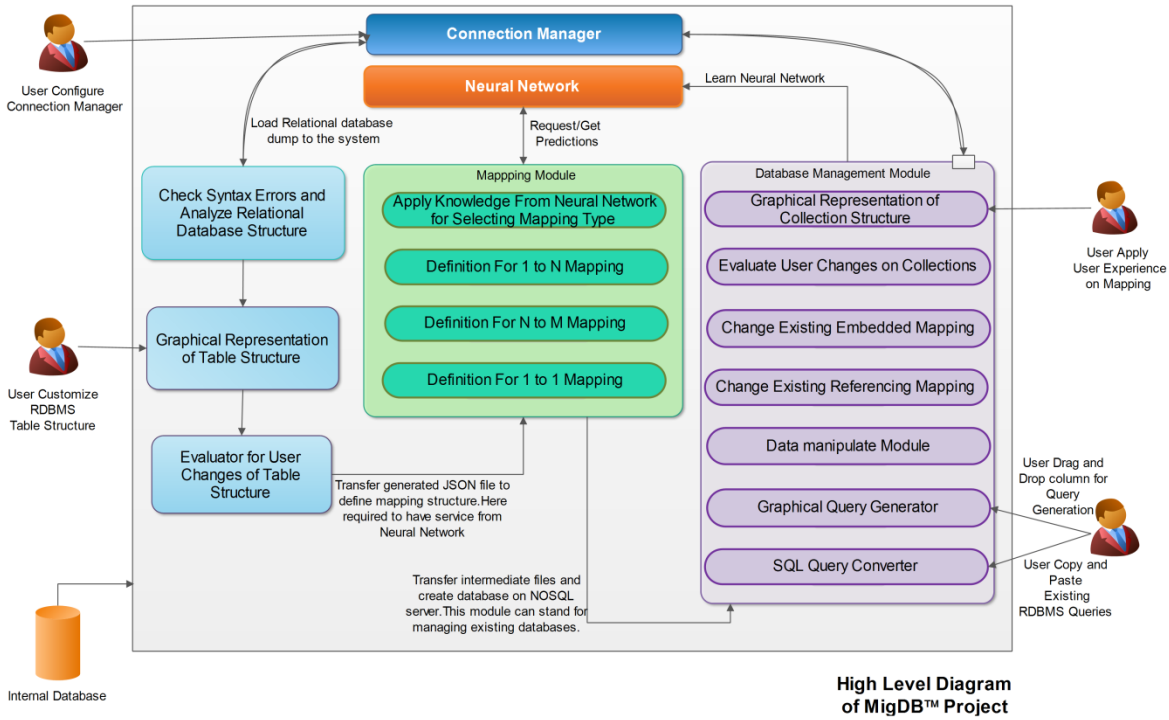

Figure 1 : System Diagram

The main objective of the system is to facilitate to users to easily convert Relational database into Mongo database along with MySQL table structures, data, queries and relationships without requiring proper knowledge on JSON based or Mongo commands. In order to complete the main objective, system divided into three categories as Relational database handling module, Mapping module and Management module and intended system uses a stage-wise approach while interacting with the user in the process.

At the beginning of the migration process system requires the SQL database to be migrated. The SQL database can be provided either by uploading the MySQL dump file to the system

or selecting the database to be migrated by using connection manager. The connection manager is sub objective of the Relational database handling module. It has capability to establish a connection with a local machine or remote server. By using connection manager system can connect to the both MySQL server and Mongo server. In here system is going to validate dump file, if the user uploaded MySQL dump file. After those processes, system is going to analyze dump file to extract MySQL table structure, data and relationships. At the completion of analyzing process, SQL DB Analyzer module creates JSON file for identified table structures, relationships and data. Using the JSON file SQL DB Modifcation Evalator represent MySQL table structures graphically and user has rights to make the changes to it such has remove tables/columns they do not desire to be migrated. System is going to be validating those changes done by user and if there is an error system will notify it to the user. Otherwise it will update JSON file with user changes. This updated JSON file forwarded to Mapping module for relationship mapping. Mapping module consists with three sub modules for perform relationship mapping. Each sub modules analyze the JSON file and if the table structure contains any relationships, suitable module is going to be mapped it an appropriate manner. In here system create another JSON file for stored mapped relationships. One-to-one module embeds data and one-to-many and many-to-many modules use neural network to get predictions about how to selecting the most suitable mapping option. After comparing neural network response with its own evaluations and those modules uses embedding or referencing appropriately. At the completion of the Mapping module, system creates Mongo database with relationships using both JSON files. The management module of the system enables users to modify the way that relationship mapping by using their own experience and also they can make changes to the collections. And there are another function under this module for manipulate the data within the document. The system support users to generate MongoDB queries using graphical query generator module and there is another advanced module for convert MySQL statement into Mongo statement easily and instantly.

2.1 Product Perspective

Along with gaining popularity addressing the issues in the relational databases and MongoDB being one of the most popular NoSQL database, there are a few systems available for the purpose of migration of an existing relational database to a MongoDB database. Some of these available products are very popular but still have some issues in particular criteria.

Pelica Migrator and NoSQL Manager for MongoDB Professional are some well-known products that enable users to migrate an existing relational database into MongoDB. Pelica migrator allows users to migrate from relational databases including MySQL, Oracle, SQL Server 2008, PostgreSQL, DB2 and Sybase into MongoDB. The tool migrates relational database relations/tables into embedded collections or documents in MongoDB and also replicates the relational database data into MongoDB. Pelica Migrator also allows users to select/ unselect tables and columns to be migrated. NoSQL Manager for MongoDB tool is capable of migrating MS SQL Server and MySQL relational database tables together with data into MongoDB. But the issue with these two tools is that they do not consider table relationships when migrating, meaning they are incapable of migrating table relationships into MongoDB. The Pelica Migrator also does not consider foreign key constrains when user unselects tables and columns to be migrated and therefore allows user to remove a table or a column which could be referred as a foreign key in another table. However, the intended system 'MigDB' is capable of migrating table relationships in the most appropriate way and also evaluates user changes for any violation of constrains when user removes columns or tables that they do not wish to be migrated.

RoboMongo, UMongo and Admin Mongo are some MongoDB management tools and enable users to manipulate existing MongoDB databases and to create and modify collections and documents via a graphical user interface. But these tools still require JSON or MongoDB knowledge from the user to some extent where 'MigDB' does not require JSON or MongoDB command knowledge from the user, provides graphical user interface to generate MongoDB commands and therefore can be used by even a novice in MongoDB.

Query Mongo and Klaus.dk are popular query translators which can translate MySQL queries into MongoDB commands but these applications are capable of translating only the select queries and does not support table joins within select queries as well. Nonetheless, the proposed system 'MigDB' has the ability to translate create, alter, drop table, insert, update, delete record and select statements along with support for table joins.

2.1.1 System Interfaces

- Connection Manager would need connection to MySQL server on local or remote server and MongoDB server
- SQL DB Analyzer module would require the library TGSqlParser in analyzing the MySQL database

- One-to-one mapping module would require connection to MongoDB server
- Graphical Query Generator module would need access to internal SQLite database in order to get MongoDB syntaxes.

2.1.2 User Interfaces

- Connection Manager

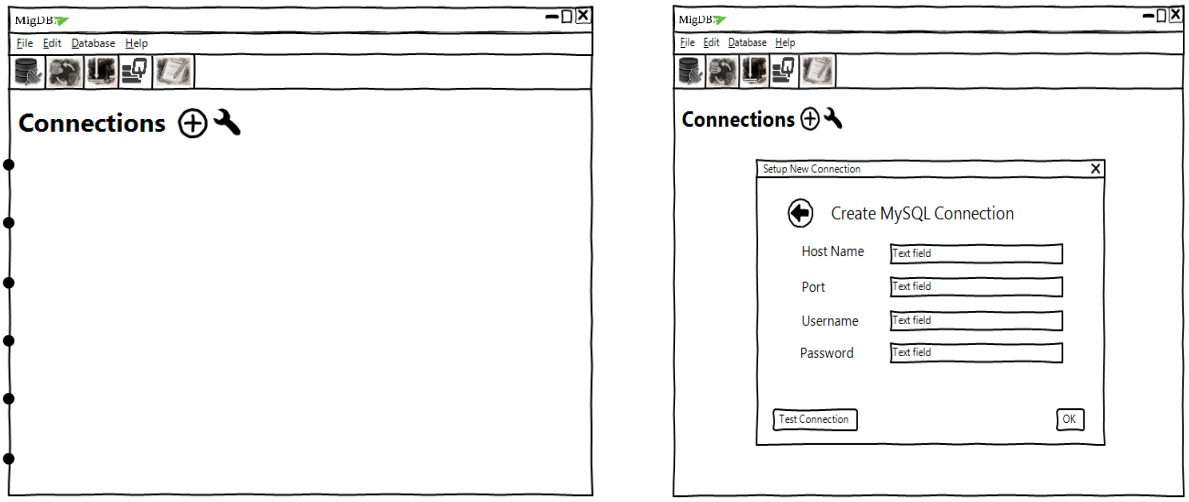

Figure 2 : Connection Manager UI

- One-to-one mapping

  One-to-one mapping module does not have an user interface since it is an internal module

- Graphical Query Generator

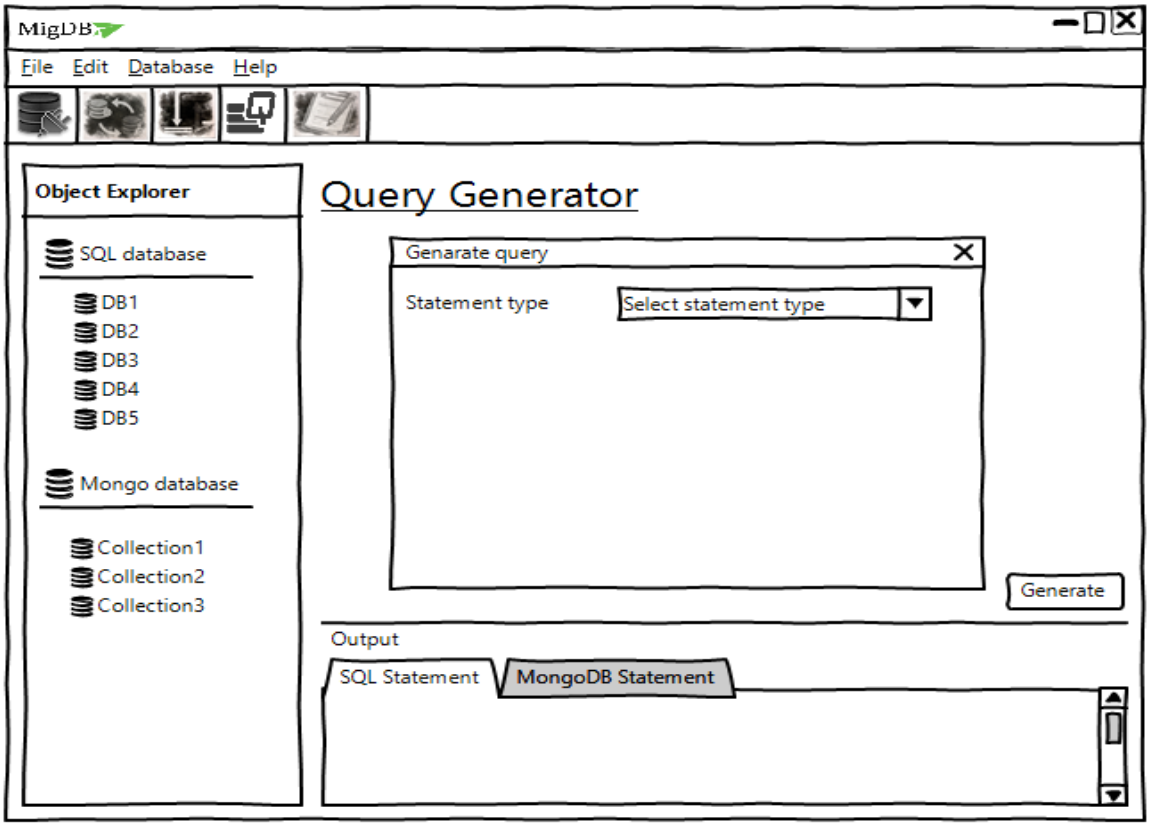

Figure 3 : Graphical Query Generator UI 1

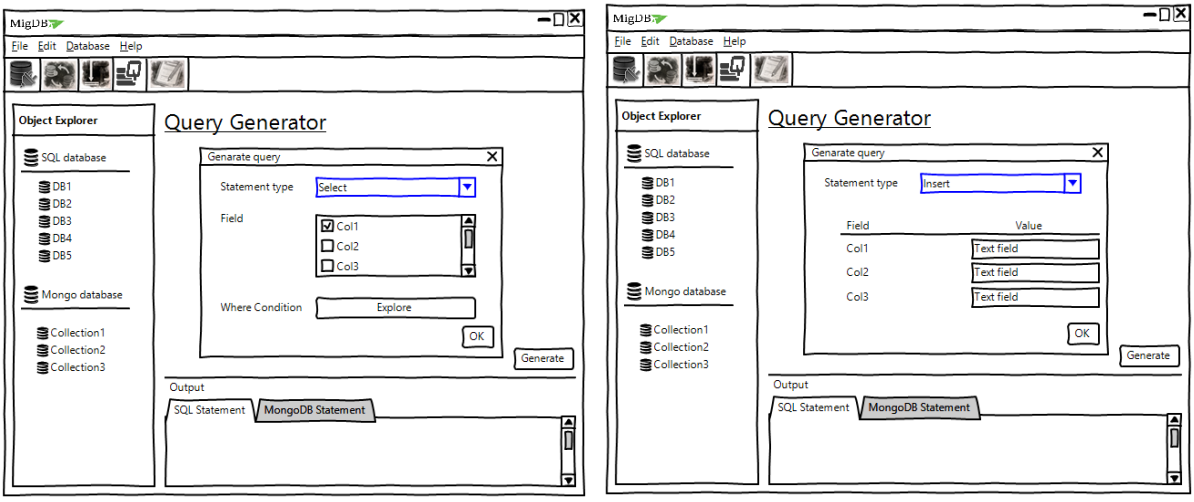

Figure 4 : Graphical Query Generator UI 2

The system uses step by step process to achieve the main goal of the system and above figures represent the basic sketches. Provided sketches give an idea for the developers on how to design the each processes in the wizard.

2.1.3 Hardware Interfaces

- Wifi router or dongle

- Service providing server
- Laptop/Desktop computers which are configured with OS such as Linux, Mac OS X or Windows

2.1.4 Software Interfaces

- Java JDK 8 is need to be configured in the machine
- MongoDB is need to installed on the machine an Mongo server need to be run to startup the machine
- Application is developed using Eclipse Mars.2 (4.5.2) and Scene Builder 8.0 is used for UI designing
- SQLite is used as an internal database to store MongoDB statements and other configuration files

2.1.5 Communication Interfaces

A modem or a router will be required to download the system as 'MigDB' is downloaded via the web. And internet connectivity is also used to access neural network services as it is hosted on the server.

- Dialup modem
- Broadband internet
- Dialup or Broadband connection with an internet provider
- Web hosting server

2.1.6 Memory Constraints

- For migration process it will need free space of the same size of the database to be migrated

2.1.7 Operations

- User downloads the 'MigDB' setup via the web application provided
- Install the application
- At the beginning user need to provide hostname, port, username and password for the MySQL and hostname, port and schema if needed, for the MongoDB to establish connection between local server or remote server.
- User can select MySQL databse which wish to migrate from the given database list or he/she can upload MySQL database dump into system.

- In case the uploaded file is validating by system and if there are errors, system will notify it to the user. Otherwise it'll analyze the dump file.
- Those analyzed records stored in JSON file.
- Using JSON file system will represent the MySQL table structure and user can do modifications for it.
- Those modifications are evaluated by the system and if there are errors, user get notified about it. Otherwise update JSON file with the changes done by user.
- Using this JSON file One-to-one mapping module embed relationships into document and create another JSON file for stored those records.
- One-to-many and many-to-many modules do same thing by using knowledge of the neural network. Those modules decide either embed or referenced.
- After mapped all relationships, system will create Mongo database with relationships.
- Using manipulating module, users can make changes to the existing mongo collection and manipulate mongo database.
- Using graphical query generator, users can generate Mongo statements graphically.
- Using SQL query converter, users can convert existing SQL statement into Mongo statement.

2.1.8 Site Adaptation Requirements

- User needs to access the web by the use of a web browser for downloading the system setup
- User needs to provide MongoDB and MySQL database connections
- User interfaces are available only in English language

2.2 Product Functions

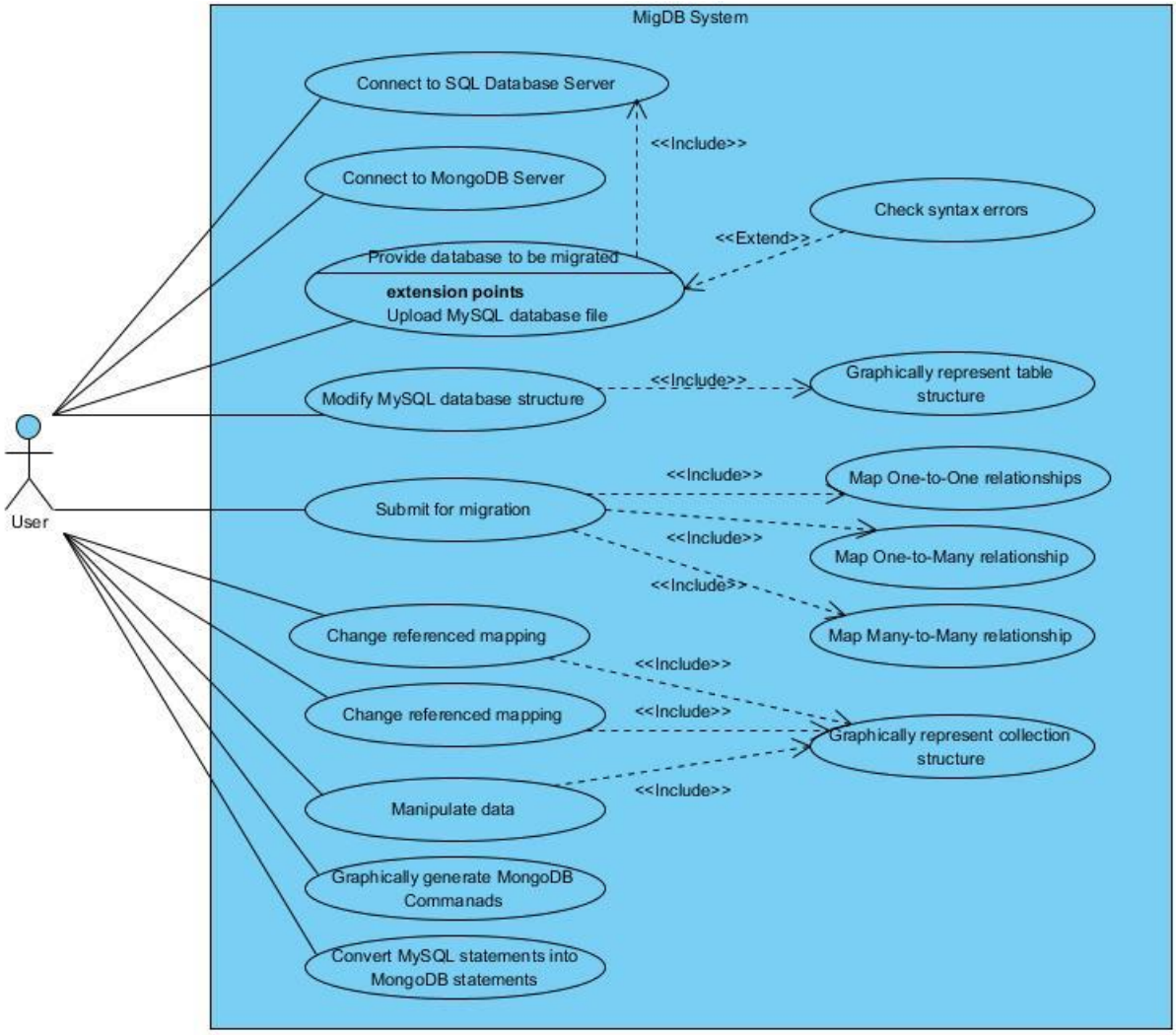

Figure 5 : Use Case Diagram

Use Case	Connect to SQL Database server
Description	Use case describes how the system make connection with MySQL database server
Pre-Conditions	User has provided a username, password, hostname and port
Post-Conditions	User inputs save and make connection successful
Primary Actor	User
Main Success Scenario	1. User click add connection button at the beginning 2. Give connection name 3. Select MySQL connection button 4. Give hostname, port, username and password 5. Click ok button

	6. Use case ends with saving details into internal database
Extensions	5a. If user give any invalid value for a parameter, the system display an error message

Table 1 : Use Case Scenario 1

Use Case	Connect to MongoDB server
Description	Use case describes how the system make connection with Mongo database server
Pre-Conditions	User has provided a hostname and port
Post-Conditions	User inputs save and make connection successful
Primary Actor	User
Main Success Scenario	1. User click add connection button at the beginning 2. Give connection name 3. Select MongoDB connection button 4. Give both hostname and port 5. Click ok button 6. Use case ends with saving details into internal database
Extensions	5a. If user give any invalid value for a parameter, the system display an error message

Table 2 : Use Case Scenario 2

Use Case	Provide database to be migrated
Description	Use case describes how the system get database file which wish to migrate
Pre-Conditions	1. MySQL connection should be established 2. User need to select database from database list or upload database file into system
Post-Conditions	If user upload database file, system check syntax errors for it
Primary Actor	User
Main Success Scenario	1. Include :: (Connect to SQL Database server) 2. System listed down all the available database 3. Use case end with user select database from list or upload database file to system
Extensions	3a. Check for syntax errors if the user upload database file into system

Table 3 : Use Case Scenario 3

Use Case	Check syntax errors
Description	Use case describes how the system validate uploaded database file
Pre-Conditions	1. User need to be upload database into server

Post-Conditions	Give error or success message
Primary Actor	User
Main Success Scenario	1. Use case begins with user upload database file into system 2. System check for syntax errors 3. Use case end with giving notification if the file validated or not

Table 4 : Use Case Scenario 4

Use Case	Map one-to-one relationships
Description	Use case describes how the system maps one-to-one relationships within the MySQL database to be migrated into MongoDB
Pre-Conditions	System has analyze database structure and create JSON file for store analyzed records
Post-Conditions	One-to-one relationships within the database has been embed into document and the create JSON file for store documents
Primary Actor	User
Main Success Scenario	1. Use case begins with, update of JSON file according to the user changes 2. System analyze JSON file and recognizes table objects with one-to-one relationship 3. Then system embed objects into document 4. System create another JSON file for store documents

Table 5 : Use Case Scenario 5

Use Case	Graphically generate MongoDB commands
Description	Use case describes how the user can generate MongoDB statements using graphical way
Pre-Conditions	1. MongoDB database connection should be established 2. System is up and running
Post-Conditions	The corresponding MongoDB statement is generated
Primary Actor	User
Main Success Scenario	1. Use case begins with the user drag and drop collections into workspace 2. User has to click on the collection 3. System popup option dialogue 4. User has to give inputs 5. Click ok button and after click generate button 6. The system analyze the statement and calls the corresponding conversion method 7. Use case ends with the system output the generated MongoDB command

Table 6 : Use Case Scenario 6

2.3 User Characteristics

The target audience of 'MigDB' is developers and database administrators who wish to migrate their existing MySQL databases into MongoDB. But the system can also be used by novices due to the system not requiring any prior knowledge on JSON or MongoDB and can also be used by the novice users to get familiar with MongoDB commands due to its functionalities like Graphical Query Generator.

2.4 Constraints

- The system would be available only in English language
- There should be internet connectivity
- MongoDB should be installed in the user's computer
- Implementation is done using Java 8

2.5 Assumptions and Dependencies

It is assumed that the user has English language literacy to some extent due to all functionalities being presented in simple English. User should be familiar with databases and table relationships in general.

2.6 Apportioning of Requirements

The document gives a general outline discussing about the purpose, scope and providing a brief overview of the system and then provides an overall description of the system intended for a non-technical user and finally discusses the requirements and constraints in system thoroughly in section one, two and three respectively.

The main three modules Relational Database Handling module, Mapping module and Management module are developed in the mentioned order. Initially since the project focuses on migration the first two modules are developed and then third module is developed while also maintaining non-functional requirements.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

- Connection Manager

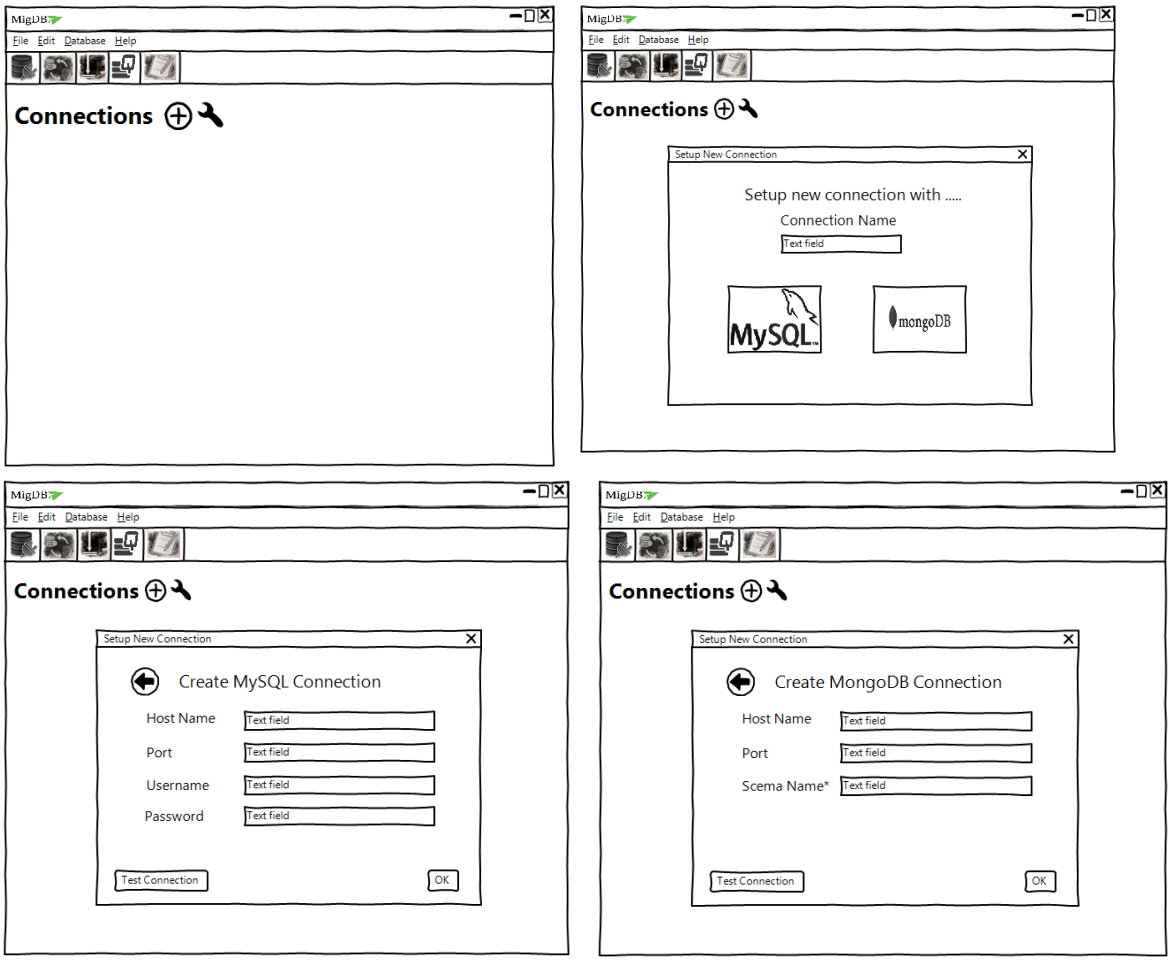

Figure 6 - Connection Manager UI

The above sketches represent the Connection manger. This connection manager has capability to establish multiple database connections with MySQL or MongoDB in a local machine or a remote server. At the beginning of the process, system ask users to create database connection. In this step user should give valid values for both MySQL and MongoDB and users can test for connection before saving it. After that process system stored those parameters into internal database. If the use need to create another connection object he/she can make new connection. If the users need to update existing connection also system

provides mechanism for doing it. The connection manager uses by Management module and Relational database handling module.

- *One-to-One Mapping module*

This module has no interfaces, because of that it is internal to the system. Main purpose of this module is embed one-to-one relationships in to the Mongo documents. This section use JSON file created at Relational database handling module analyze it to findings objects which has one-to-one relationships. Final outcome of this module is embed one-to-one relationships to documents. In this section it will create another JSON file to store analyzed records.

- *Graphical query generator*

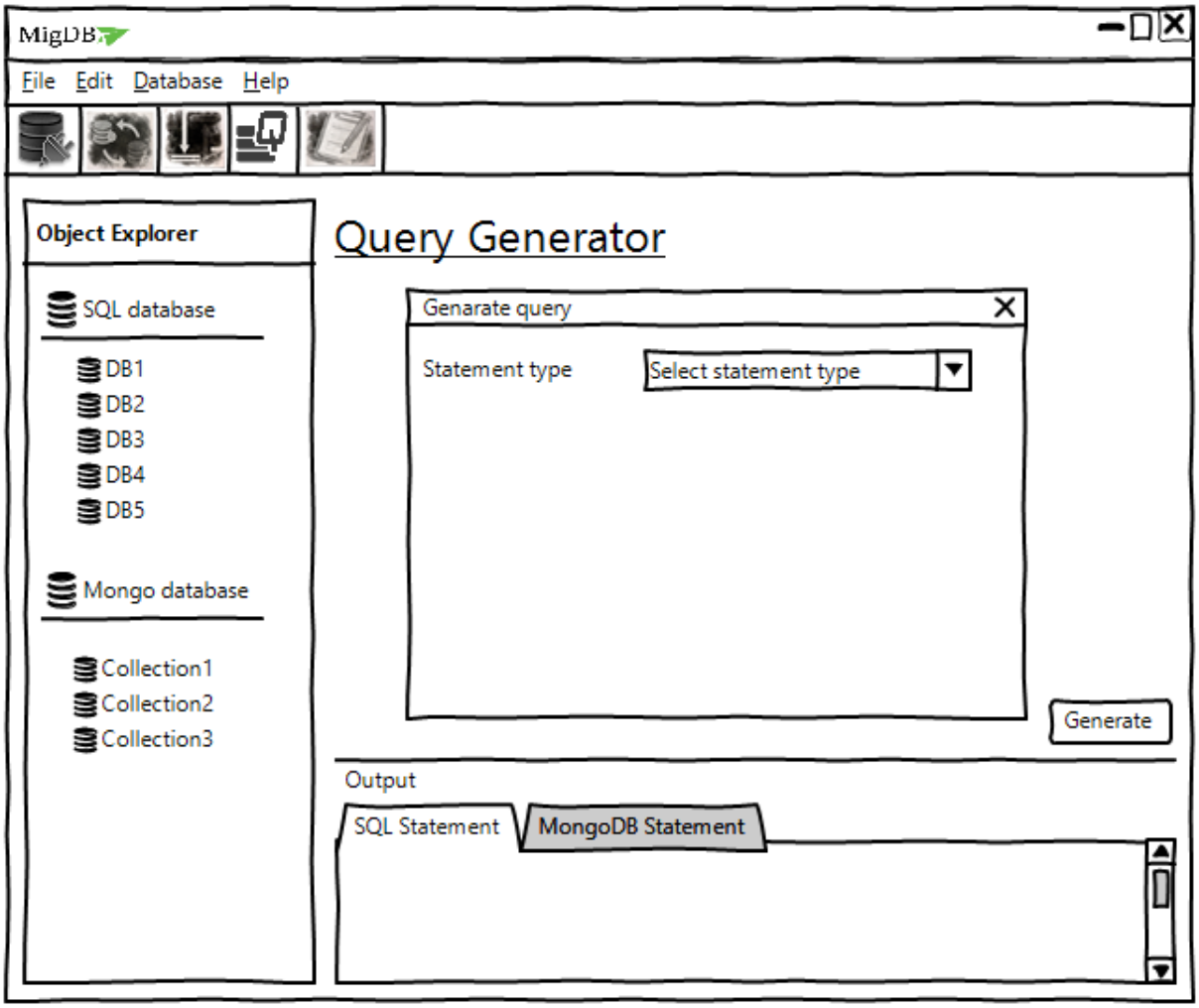

Figure 7 - Query Generator UI-1

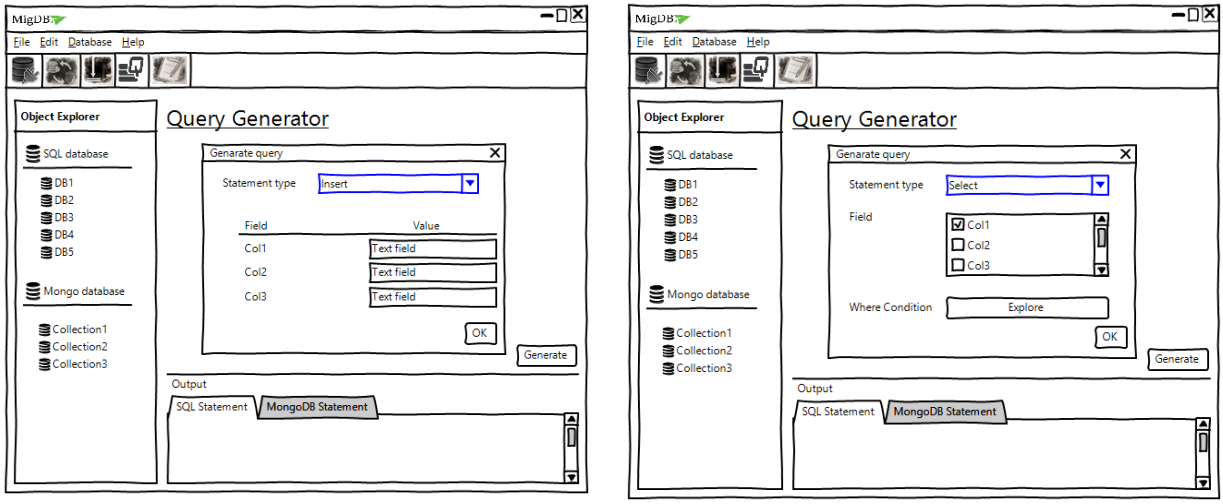

Figure 8 - Query Generator UI-2

The above sketches represent the Graphical query Generator. Using this feature users can generate the Mongo statements by using graphical interface. Connection manager is pre-requisites for this module and all the MySQL databases shown in here. User can expand those databases. Under the database objects, all the collections represent. Users can drag and drop collections into working surface. And after that by click on the collection it will popup window to select statement type. According to the statement type user interface is going to change. The above figures show only insert statement and select statement. In this point, system ask from the user to make necessary changes to the provided fields. After that process user can get the generated MongoDB statement by pressing generate button.

3.1.2 Hardware Interfaces

- Wifi router or dongle

  A router or a dongle is needed for internet connectivity in order to download the software and to request hosted neural network services
- A Server

  A server is required for hosting of the web application and the neural network web service
- Laptop/Desktop computer

  A computer with a good processor and necessary disk space

3.1.3 Software Interfaces

The Eclipse Mars.2 (4.5.2) is used as the IDE. It's a full java IDE with advanced feature which enables to test, debug and interact with repositories. JavaFX is used as the software platform for creating desktop applications. Scene Builder as a tool which running on JavaFx

platform and it enables to quickly design rich JavaFX applications and it can be configured with Eclipse IDE.

- Latest Java JDK 8
- Eclipse 4.4 or greater with e(fx)clipse plugin
- Scene Builder 8.0

3.1.4 Communication Interfaces

Users can get system via the web. Internet connection is needed for downloading application. Therefore, a modem or router will be required to establish internet connection. Also system requires internet connectivity to access neural network services from the hosted server location.

- Dialup or Broadband connection with an internet service provider
- Web hosting server

3.2 Classes/ Objects

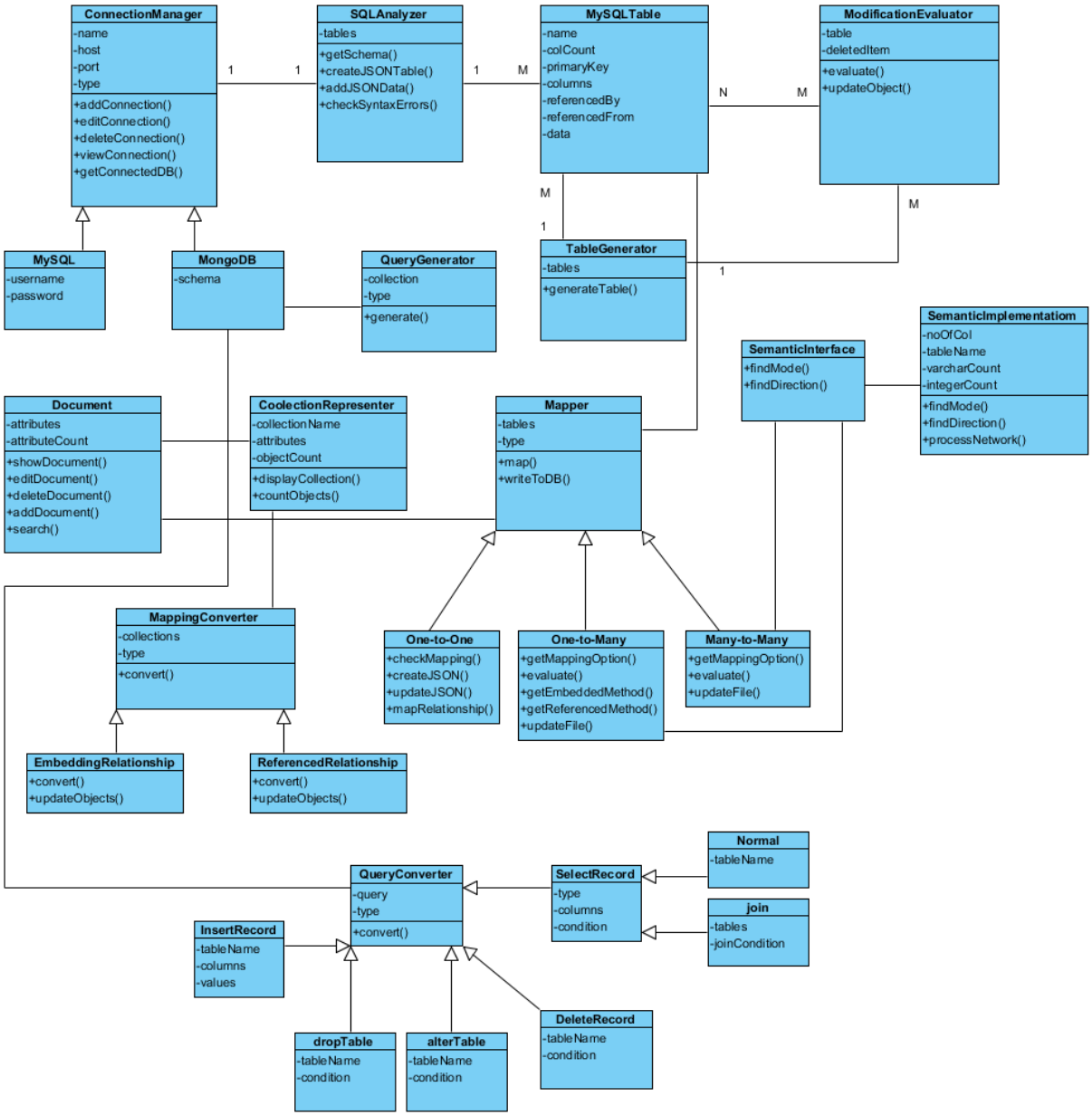

Figure 9 - Class Diagram

3.3 Performance Requirements

The system should be able to handle all the operations without freezing under required hardware and software requirements.

The cloud server can handle up to 1000 simultaneous users without freezing.

3.4 Design Constraints

Java should be used as the implementation language since it is platform independent.

SQLite is used to store configurations, constraints and MongoDB syntaxes since it should be internal to the system.

3.5 Software System Attributes

3.5.1 Reliability

The term reliability is the ability of the software system to provide it's functionality to consistently perform according to its specification/for a specific time period. The system 'MigDB' does not fail more than twice a week. And the system is 95% free of technical errors. Therefore use efficient and appropriate methods to perform all the operations through out the system. Final product of the system fully tested on different platforms and appropriate software and hardware should be used to minimize the issues.

3.5.2 Availability

Availability defines the proportion of time a system is serviceable to total time it is required or estimated to function. It can be measured as a direct proportion or as a percentage. The system 'MigDB' targeted to achieve high availability around 99% to guarantee a higher level of user satisfaction by reducing the downtime as much as possible.

3.5.3 Security

System provides high security with using java security api's to communicate with the server side application. And also web application use security mechanism such as avoid SQL query injection, use encryption method for confidential data, use secure payment gateways for perform an payment methods.

3.5.4 Maintainability

Maintainability of a system is describes as the degree to which an system is repaired, quickly and easily add new features and well understaood. The system 'MigDB' use coding standards, Object oriented programming skills, web services and also users can update the software.

4 Supporting Information

4.1 Appendices

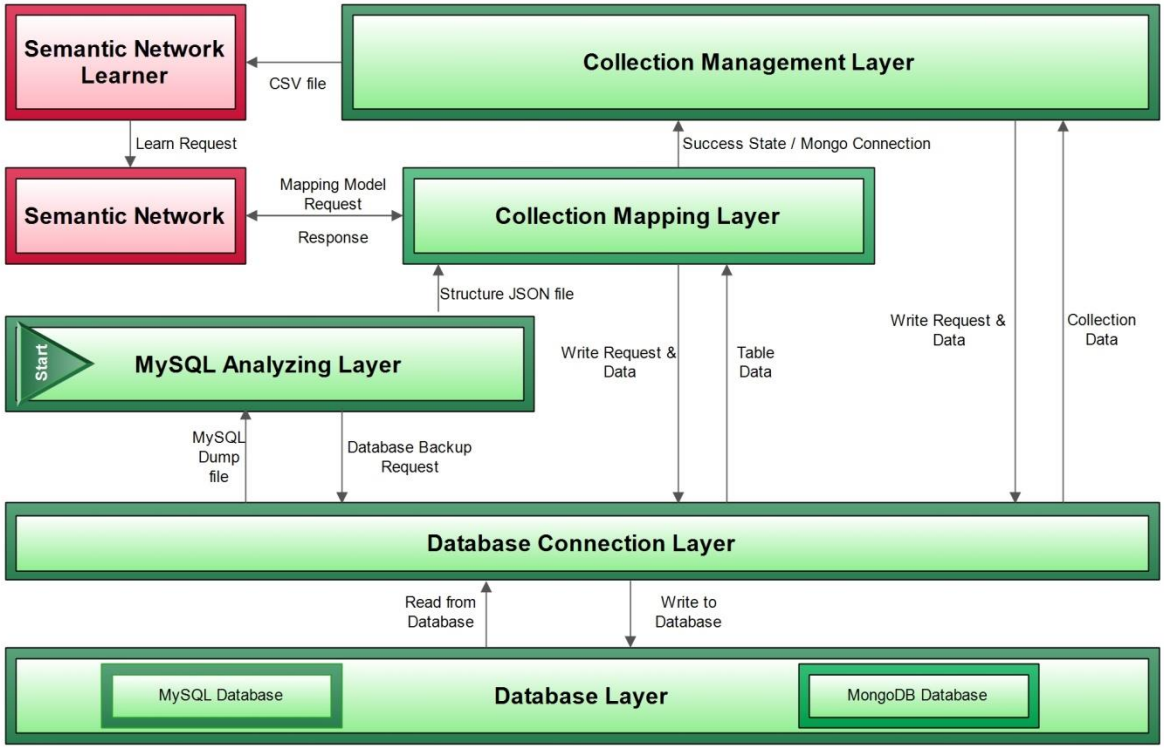

Figure 10 : System Architecture

References

[1] Margaret Rouse, "Reliability, Availability and Serviceability (RAS)", *whatis.techtarget.com*, March 2011. [Online]. Available : http://whatis.techtarget.com/definition/Reliability-Availability-and-Serviceability-RAS [Accessed: April 1, 2016].