|| Volume 2 ||Issue 1 ||JANUARY-MARCH 2017|| ISSN (Online) 2456-3293

OPEN ACCESS INTERNATIONAL JOURNAL OF SCIENCE & ENGINEERING

GUI BASED QUERYING AND MIGRATION APPROACH FOR MONGODB

Mr. M. B. Jadhav[1], Prof. R. R. Badre[2]
P.G. Student, Computer Science & Engineering, MIT AOE, Alandi, Pune, Maharashtra, India. [1]
Assistant Professor, Computer Science & Engineering, MIT AOE, Alandi , Pune, Maharashtra, India. [2]

Abstract: With the exponential growth of Big Data on the transactional side of the web architecture, the use of NoSQL databases has seen a considerable amount of growth in the recent years. Going along, one of the keys aspect that keeps every database developer concerned is the ability to communicate dynamically and exchange data between the traditional RDBMS, Hadoop and the No SQL databases. Moreover, the advancements in the application developments also increases the overhead of the developer in writing complex yet optimized queries. There are a number of applications available in the market that focuses on the Migration activities as also we have some applications which provides auto complete features to ease the development part. But this again leaves you managing a bulk of tools which is a very tedious task working on critical application. In this model, we would be building a central web based graphical tool for exchanging data between MongoDB, Hadoop and RDBMS. Additionally, this system will provide the first ever completely optimized EMS with graphical querying features. Now let it be a simple or a complex query or an aggregation function or an aggregation Map Reduce, you need not write a single line of code or command to get things done.

Keywords: *MongoDB, RDBMS, Migration, Relational Model, Schema*

I INTRODUCTION

There exist two major problems in the world of databases that we would try to take care of through the implementation of this project.

A. No GUI Tool

Though we have n number of client tools available for the databases, there is no tool which provides a complete GUI for working with data. The user is expected to have knowledge of databases and should be well versed with techniques involved in fetching and processing the data. This results in the application developers spending most of their time with database developers to make sure that they have the perfect query based on their specifications. Moreover, with more aggregate functions into picture, increases the complexity of writing optimized methods for processing the data. This makes it even more difficult for developers to work with new trending databases like MongoDB which do provide extensible features but do not follow the traditional SQL framework. In this paper we aim towards building a web based GUI tool which will eradicate the necessity of knowing the syntaxes by providing step wise iconic representation of different methods for processing the data. Moreover, the integrated library of the tool will also provide a facility to auto correct all errors. So now even if you are zero in databases but atleast know what you are looking out from your data, this tool will enable you to fetch exactly what you need in the most optimized way without any glitches.

B. No Universal Data Migration Tool:

With advent of Big Data, managing data using traditional RDBMS becomes really complex and tedious work. It has been observed that all the features of the RDBMS like the schema, Tabular Structure, SQLs, Joins and ACID have turned out to be limitations restricting the use of these databases while handling Big Data. NoSQL databases like MongoDB on the other hand make it relatively much easier to handle Big Data on the transactional front. As a result of this, in the recent years the industry has seen a lot of data migration happening from SQL databases to NoSQL databases. This migration however introduces new challenges for the developers of handling different tools for exchange of data between different platforms. Moreover, it also introduces the overhead of transforming the queries from one form to another to suit the syntaxes of different data storage platform which use different languages for processing the data. Through the implementation of this paper we propose to build a tool which will enable easy data migration between MongoDB, MySQL and Hadoop. With a complete GUI nature of the tools we are looking forward to deploy the migration process with simple click functions without actually writing a single line of function, query or any sort of code. Also this tool will be aiming at building a connect environment from where any query can be converted internally to an equivalent MongoDB function and can fetch the required data from the MongoDB databases without the need of changing the query implementation from the programming end.

II MOTIVATION

Working with the Database Clients which have the auto complete features enabled, does not actually give the privilege of writing efficient queries without actually knowing anything out of it. The developer needs to atleast know some part of the command so that he can leverage the auto complete feature. It is good to use for the traditional databases wherein we are well versed with all the options that we have got, since we are using the SQL from over 3 decades now. But for new emerging databases like MongoDB, the scenario is a bit different. MongoDB introduces new set of efficient functions in each of its release which if used in the right potential can help processing data in a very optimized way. But because of lack of awareness of the new introductions or because of being habituated to use the common functions, the developer does not make extensive use of the features available. On the other hand, a tools like MS Paint, does not keep you thinking what you can do on a canvas but rather gives you a complete list of available options to explore so that you can paint your imagination with getting into the complexity of Computer graphics. So if this can be done on a graphical platform then why not implement something similar for the database platform to make life with databases easier.

III TAXONOMY AND TERMINOLOGY

A. NoSQL Databases

To handle the problem of Big Data we came up with a family of databases that

1) Supports Structured as well as Unstructured Data.
2) Stores data in Flat File System
    • Data is stored in Binary Format
3) Can support anything as long as your application can understand it
    • For Example, you can write the value of the field AGE as either 24 or twenty-four or 20 + 4
4) Does not have support to the concept of Data Types
5) Can scale Horizontally
6) Use functions to query the data. So that working with Flat Files becomes easy.
7) Will embed all properties of an entity with one object itself i.e. have embedded objects. This eradicates the need of maintaining Joins as Joins become expensive with the increase in number of Tables.

These Databases are called as NoSQL Databases, popularly called as Not Only SQL Databases. These Databases do not follow any properties of the Traditional RDBMS. Based on how they store and Retrieve Data, they can be broadly classified as

1) Key Value Pair Store Databases
    • Popular in this category is Amazons Dynamo
2) Columnar Store Databases
    • Popular in this category are Googles Big Table and Facebooks Cassandra.
3) Document Store Databases
    • Popular in this category is 10 Gens [now known as MongoDB University] MongoDB
4) Graph Based Databases
    • Popular in this category is Neo4j

B. MongoDB:

MongoDB is the leading NoSQL Database available in the Market Today. MongoDB is derived from a Latin word called as Humongous, which means enormous or huge. So from its name itself MongoDB makes a clear statement of being enormous with storage as well as processing capabilitie

1) Popularity of MongoDB:

1) MongoDB was awarded as the Database of the year 2014 and 2015. With this MongoDB became the first ever Database to achieve this feat consecutively
2) MongoDB Stands out to be the top most ranked Database as compared to all NoSQL Databases and ranks 4th as compared to all the Databases available.
3) MongoDB is deployed on the production deployments of more than 70 percent of the Fortune 500 Companies.
4) MongoDB has been recently deployed on to the Small Cap and Mid Cap Companies as well.

2) MongoDB Features:

1) MongoDB is a Distributed, Document Oriented and Open Source Database.
2) Mapped with a SQL Database, Collections in MongoDB resemble to Tables in SQL Databases and Documents in MongoDB resemble to Records in SQL Databases.
3) MongoDB is an Object Oriented Database that uses JavaScript functions and syntaxes to process results out of the Collections.
4) In MongoDB all the records are organised into JSON Documents where every Document is treated as an Object.

5) MongoDB can be used for any Domain and any Kind of Application which gives it one more advantage overs its counter-parts like Cassandra and Dynamo.

6) It is a Cross Platform Database that can be installed on any Operating System may it be Windows or Linux or any other OS. Moreover, only the installation part of MongoDB is different for all the OS, but once you get into the Mongo Shell, everything is the same irrespective of the backend OS.

7) MongoDB has one of the finest integrations with almost all available Programming Languages starting right from the most basic ones like C and C++ to languages like Java, PHP, Pearl, Ruby etc.

8) MongoDB provides Automatic Scaling around a concept called as Sharding.

9) It also provides High Performance, High Availability and Automatic Failover around a feature called as Replication.

10) MongoDB provides full support to Indexing and implements Aggregation in 3 different ways of Pipeline, Standalone and Map Reduce.

11) It has open integrations with popular BI tools like Pentaho and Tableau.

12) It has the finest integration with Hadoop, providing the most powerful platform for Data Analysis.

IV RELATED WORK

F. Matthes and C.Schulz in [5] state that ”Tool-supported one-time process which aims at migrating formatted data from a source structure to a target data structure whereas both structures differ on a conceptual and/or physical level”.

According to ”Girts Karnitis” and ”Guntis Arnicans” in [1] Data Migration is a combination of 2 steps wherein we restructure the data from the source to meet the specifications of the target system and secondly initiate the transfer from the source to the destination. Several methods including but not restricted to Schema Conversion, meta-modelling, ETL and automated data migration approach deal with these steps. There are 2 levels in which data is available in a RDBMS viz. Physical Level and Logical Level. Different tree building algorithms are used to define the logical hierarchy that exists in the data.

”Gansen Zhao”, ”Weichai Huang”, ”Shunlin Liang” and ”Yong Tang” in [2] define MongoDB in to a Relational Model. The Authors define two levels of defining the schema. One of which is the Micro Level in which the Schema of MongoDB Collection can be considered to be fixed and the other one is the Macro Level in the Schema of MongoDB Collection would vary for any two given instances of time. So if we want to define the Schema of a MongoDB Collection, then considering the Micro Level we can state that the Relational Model of MongoDB is MongoDB Collection maps to Table in the Relational Model. The Schema is composed of keys present in the document the would define the structure of all tuples. Every value will be corresponding to the value of a key which will be null if the key is not present in a document. There can also exist Parent-Child relations between tables which can be identified by the Primary and Foreign Keys which will define the Keys in the Main Document and the Keys for the Sub Documents [4].

”Thalheim” and ”Wang” [3] state that in order to migrate the data, one needs to have a thorough understanding of the data source such as data availability and data constraint since different data sources are designed using different modelling semantics. Mongify is a tool that enables data translation from SQL databases to MongoDB. It provides integration with MySQL, PostgreSQL, SQLite, Oracle, SQL Server and DB2. It works well with all versions of MongoDB. MongoBooster is a Cross Platform GUI tool for MongoDB working on a shell centric platform providing in-place updates and integration with Moment.js.[7] It supports the ES6 syntax providing a true intelligence experience. It provides support for mongoose like fluent query builder API. API which enables you building queries using chaining syntaxes instead of simple JSON Objects. Writing more concise and readable MongoDB scripts becomes easy with the built-in support for block variable scoping, arrow functions and template strings.

RoboMongo [6] does not emulate the shell of MongoDB rather it simply embeds the same environment and engine available with the Mongo Shell. It is currently supports the MongoDB version 3.2. RoboMongo executes the code in an internal VM based on JavaScript rather than simply analysingthe semantics of the code, giving the user an auto complete feature at runtime, adding features that cannot be obtained otherwise using the static methods.

NoSQL viewer stands out to be a free GUI based client for NoSQL databases like MongoDB, Cassandra, Couch base, Couch DB and Hbase. It enables users to simultaneously perform CRUD operations from one single platform eliminating the overhead of using multiple tools for different databases. It provides easy yet powerful, high performance migration functionality between any supported Big Data databases.

MongoDB Connector for Hadoop enables the integration between the 2 most powerful data storage systems from the OLTP and OLAP sectors. It allows the ability to use MongoDB and Hadoop as sources and destinations for data transfers.

Database Master is easy to use database querying, administration and management tolls that provides a consistent user interface with modern styling and interface. Is simplifies the process of managing, monitoring, querying, editing, visualizing and designing relational and NoSQL database systems. It allows users to execute extended scripts in SQL, JSON and C Sharp providing all database objects like tables, views, procedures, packages, columns, indexes and triggers.

V PROPOSED WORK

This project is supposed to be a blend of 3 Web Enabled GUI Applications into one single platform

1) A migration tool that can exchange data between
    - MongoDB and RDBMS
    - MongoDB and Hadoop
2) A GUI tool that will provide
    - Drag and Drop elements pertaining to
    - CRUD Operations
    - Aggregation
    - Indexing
    - Replication
    - Sharding
    - Autofill syntaxes wherever applicable
3) Administrative tool that will provide
    - One Click MongoDB Cluster Setup
    - Cluster Monitoring
    - Replica Set Setup
    - Sharded Cluster Management
    - Query Transformation

VI CONCLUSION

In this way we are looking forward to build a tool that will ease of the pressure from the Developers to understand the syntax and will help build better queries. Moreover, it will enable quick Migration activities as the tedious load of query translation will be take care of. We would be implementing the GUI tool for CRUD operations which can be extended further to support aggregations as well and the Migration tool that we would be building will support 1NF which can be further extended to support higher hierarchies.

ACKNOWLEDGMENT

I would like to thank department of computer engineering MIT AOE for providing such enthusiastic, energetic and warm environment for the development of student's creativity. I am thankful to guide Prof. R. R. Badre for providing consistent support throughout the work.

REFERENCES

[1] Girts Karnitis and Guntis Arnicans Migration of Relational Database to Document-Oriented Database: Structure Denormalization and Data Transformation. 7th International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN).

[2] Gansen Zhao, Weichai Huang, Shunlin Liang, Yong Tang Modelling MongoDB with Relational Model. 2013 Fourth International Conference on Emerging Intelligent Data and Web Technologies.

[3] B. Thalhein and Q. Wang Data Migration: A theoretical perspective . Data and Knowledge Engineering, vol. 87, pp. 260-278, 2013.

[4] Aryan Bansel Horacio GonzalezV elez Adriana E. Chis Cloud-based NoSQL Data Migration. 2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing.

[5] F. Matthes and C. Schulz Towards an integrated data migration process model. Software Engineering for Business Information Systems (sebis), 2011

[6] *www.robomongo.org/.*

[7] *www.mongobooster.com/.*

[8] *www.mongodb.com/cloud* .

[9] *www.mongify.com* .

[10] D. Lee, M. Mani, F.Chiu, and W. Chu translating relational schemas to XML schemas using semantic constraints. proceedings of the 11th CIKM, 2002, pp.282-291.