MySQL and MongoDB as Methods of Storing Data

Deirdre Connolly

Computer Systems

Department of Computer Science

Cork Institute of Technology

Rossa Ave, Bishopstown, Cork, T12 P928, Ireland

deirdre.connolly1@mycit.ie

The goal of this paper is to examine the use of MySQL and MongoDB as methods of storing data, based on my experience of completing the assignments, the theory covered in lectures, as well as other research that I have conducted. The reason for doing so is to gain an understanding of how each method works, in order to be able to apply this knowledge practically. The comparison shall comprise of executing queries given in the assignments, with each question individually utilizing one of the two systems.

The experiment showcased where each database system had its merits, and where each was lacking. It allowed for the user to discover both systems via practical means, garnering the knowledge in knowing which method of data storage is most applicable in different situations.

I. Introduction

In today's modern technological society, there is a vast amount of data generated every single second. All of this data needs to be stored correctly for websites to function effectively. A user's login credentials are required to match to their password, which must be encrypted securely. Pictures, videos, audio, and text are constantly being uploaded, downloaded, created, destroyed, or updated. These changes need to be tracked appropriately in order for the system to be accurate and reliable. Different data storage systems go about this in various ways, depending on how they are designed and utilised.

This report shall focus on how MySQL and MongoDB store and manipulate data. This will be achieved by first familiarising myself with both systems, implementing the queries from assignments one and two. Having completed the given questions, a comparison will be carried out, noting the similarities and the differences between MySQL and MongoDB. Proposals will be made regarding the suitability of each system in regard to data storage optimisation.

This report will focus on a case study, namely assignment one and two. An overview of MySQL and MongoDB will first be given. Following that, the advantages and disadvantages of each will be examined and recorded. The applicability of each will be discussed, in reference to the type of applications that they would most be suited to. Finally, a conclusion will be drawn from my experience of using each system.

II. MySQL

The naming of MySQL is derived from "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation of Structured Query Language. It is an open-source relational database management system (RDBMS). Originally released in May of 1995, it is now used by numerous websites with a gigantic user base, including Facebook, Twitter, and YouTube. It is also used by Google, however not for its search engine, but rather it was used to build AdWords.

MySQL is based on a relational model, meaning it is composed of tables. These relational tables lead to an enforced schema. The focus is set on what the data consists of. A table is comprised of rows and columns, where a row represents records, and a column represents attributes. A single row of a table, which contains only a single record for that relation, is referred to as a tuple. The same row can appear more than once in a MySQL table; however, the same tuple cannot appear more than once in a relation. Each row has one or more attributes, known as the relation key, which can uniquely identify the row in the table.

A MySQL database uses normalization to store data efficiently. This minimizes redundancy, reduces the amount of storage capacity needed, and makes maintenance of the database easier.

MySQL allows for transactions to occur. A transaction is a set of one or more SQL statements that perform a set of related actions, that is to say, an amount of work that you want to treat as a one whole unit, rather than each individual executed statement. Transactions solve the issue of multiple users attempting to view and modify data at the same time. Each attempt at modifying the database is isolated, thus ensuring that the data maintains accuracy and consistency. If a transaction is making multiple changes to the database, either all of the changes succeed when the transaction has been committed, or else all of the changes are undone when the transaction is rolled back to its savepoint.

RDBMS utilises ACID (Atomicity, Consistency, Isolation, Durability) transactions in order to guarantee validity. The

relational model is transactional, which implies fault tolerance (*e.g.* power failure), and deterministic behaviour are provided.

Atomicity means that either *all* of a transaction happens, or *none* of it happens. Consistency means that all of your data is guaranteed to be consistent, *i.e.* any constraints that you have set will never be violated. Isolation means that one transaction cannot read data from another transaction that has not yet been completed. If two transactions are executing concurrently, each one will see the world as if they were executing sequentially, and if one needs to read data that is written by another, it will have to wait until the other is finished. Durability means that as soon as a transaction is complete, any changes made are guaranteed to have been recorded accurately.

One notable drawback of using MySQL is the lack of performance diagnostics. In addition to this, it has been found to crash when trying to deal with too many operations at a given time. While it may work just fine for small or medium applications, as the data size grows, the performance degrades. Queries take too long to execute entirely, or may time-out before completing. Unfortunately, MySQL is not designed to be scalable, and this issue is one of the leading factors in driving people to switch to NoSQL databases, such as MongoDB.

According to MySQL's own website, its customers include the likes of Uber, Booking.com, Sony, NASA, CERN, Eli Lilly, Tesla, Spotify, Netflix, YouTube, Nokia, and Twitter. However, in recent years other big names have jumped ship in favour of something else, such as Red Hat Enterprise Linux, Slackware Linux, and the Wikimedia Foundation all migrating to MariaDB. Furthermore, MySQL was acquired by Oracle in 2009, so although it has technically remained open-source, it no longer feels that way to developers using it. Many users have been left frustrated at its lack of positive change since Oracle took over.

III. MongoDB

MongoDB is a cross-platform document-oriented NoSQL database. It uses JavaScript Object Notation (JSON) and Binary JSON (BSON) documents. Initially released in February of 2009, it is used by prominent websites such as Google Search, Codeacademy, Foursquare, IBM, Uber, eBay, Cisco, and Bosch. It is written in C++, and supports application programming interfaces (APIs) in many languages, such as JavaScript, Java, and Python. The Mongo shell is a fully-functional JavaScript environment used interact with the data.

MongoDB is a hash-based, schema-less database. It does not need any pre-defined data schema, meaning every document (row) in a collection (table) could be comprised of different data. The number of fields, content, and size of a document can vary from one document to another. Unlike MySQL, there is no data definition language (DDL). This means that you can store hashes with any keys (basic data type stored as a string) and values that you choose.

MongoDB has been designed with both scalability and developer agility in mind, making it ideal for productions that are expected to grow over time. In MongoDB, the working set is stored in internal memory, enabling faster access to the data. However, this comes at a cost as more RAM will be needed. As there is no need to map application objects to database objects, meaning no ER diagrams, data can be integrated faster than in MySQL.

MongoDB utilises master-slave replication with automated failover, making it a robust system. Replica sets of the data are created and stored, and can be accessed if the original data fails, providing high availability. It uses auto-sharding, a method of partitioning that splits larger databases into smaller sections, which are faster and easier to manage. Up until version 4.0, it did not support transactions. That feature has since been included, however it is not well suited for complex transaction systems.

MongoDB is most befitting to applications using big data, for user data management, and for content management and delivery. As mentioned previously, this is the reason why the likes of Google Search and eBay are customers of MongoDB.

IV. Conclusion

Having completed the two assignments using queries in both MySQL and MongoDB, I made observations about each. Initially, the move to MongoDB was challenging. The syntax and naming concepts were new, requiring some familiarity before becoming comfortable using the system. After completing the first half of the queries, I started to become more proficient at using the query statements, knowing what to use in order to manipulate the data to show the necessary information. One area where I found MySQL to be more advantageous was the simple act in the way the data was displayed. The use of tables is more natural and readable at first glance. MySQL was more forgiving of errors, as a savepoint could be created allowing the database to be rolled back. The JOIN clause is a useful feature that is notably absent in MongoDB.

Using JSON to populate the MongoDB documents was efficient, however syntax was hugely important. Issues occurred wherein the source material contained errors, although once discovered, they were fixed relatively easily. Storing a larger amount of data was more suited to MongoDB, making use of its schema-less nature.