**A case study on different database management systems**

Sanika Patankar, Shaunak Deshpande

Department of Instrumentation and Control

Vishwakarma Institute of Technology

**Introduction**

When building a software application, one of the primary considerations to make is the database of the system. A Database Management System or DBMS is a type of software that communicates with the database itself, applications, and user interfaces to obtain data and parse it. The DBMS also contains the key instruments to govern the database. Databases come in two types, Relational databases and non-relational databases, also called SQL & No-SQL databases. A relational database is a type of data store organizing data into tables that are related to one another. Structured Query Language is the core of these systems as it is used to communicate with and manage these databases, having given birth to their second name — SQL databases. RDBMSs have a predefined schema, meaning data resides in rows (records) and columns (attributes) with a strict structure. Here, each record usually holds a value for each attribute, resulting in clear dependencies between different data points. A non-relational database is a non-tabular database that uses different data models for storing, managing, and accessing data. The most common data models are document- oriented, key-value, graph, and wide-column. As these databases aren’t limited to a table structure, they are called NoSQL. They allow for storing unstructured data such as texts, photos, videos, PDF files, and a bunch of other formats. Data is simple to query but isn’t always classified into rows and columns as in a relational database.

**Relational (SQL) Databases**

1. **MySQL**

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter,**[1]** and "SQL", the abbreviation for Structured Query Language. Originally an open-source solution, MySQL now is owned by Oracle Corporation. MySQL is a pillar of LAMP (Linux, Apache, MySQL, and Perl/PHP/Python) stack.

The community edition of MySQL is free to download. With a basic set of tools for individual use, MySQL community edition is a good option to begin with. There are other, prepaid options for Enterprise or Cluster purposes with richer functionality, but community edition is a good place to start, especially for smaller organisations with smaller budgets. MySQL’s structure and style are very plain. Developers even consider MySQL a database with a human-like language. MySQL is often used in tandem with the PHP programming language. Because they share a gentle learning curve, it’s much easier to form a team to manage your database. Also, MySQL is easy to use. For instance, most of the tasks can be executed right in the command line, reducing development steps. Along with that, MySQL is supported by most popular cloud providers such as Amazon, Google, Microsoft etc.

On the other hand, MySQL was not built with scalability in mind, which is inherent in its code. In theory, you can scale MySQL, but it will need more engineering effort as compared to any of the NoSQL databases. If a database is expected to increase substantially, this limitation must be considered. Because of this, MySQL is considered best for designing a small, web-based solution with a small volume of data. For example, when building a local eCommerce store.

1. **MariaDB**

MariaDB is a community-developed, commercially supported fork of the MySQL RDBMS, intended to remain free and open-source software under the GNU General Public License and has similar commands, APIs, and libraries as MySQL. Development is led by some of the original developers of MySQL, who forked it due to concerns over its acquisition by Oracle Corporation in 2009**[2]**.

When it comes to security, in addition to internal security and password check, MariaDB provides such features as PAM and LDAP authentication, Kerberos, and user roles. In combination with encrypted tablespaces, tables, and logs, it creates a robust protective layer for data. Above that, MariaDB publishes the related releases on each security update, keeping the security patches totally transparent. Although MariaDB originates from the MySQL engine, it got very far in terms of performance. Extensive optimization features improve thread pool management and data processing. Thus, when rows from the table are deleted, the operating system immediately accesses the free space, eliminating gaps in the tablespace. On top of that, the database management system suggests engine-independent table statistics. This feature enhances the optimizer’s performance, accelerates query processing, and helps customize data analysis.

**INSERT MARIADB CONS HERE.**

1. **PostgreSQL**

This database management system shares its popularity with MySQL. This is an object-relational DBMS where user-defined objects and table approaches are combined to build more complex data structures. Besides that, PostgreSQL has a lot of similarities with MySQL. It’s aimed at strengthening the standards of compliance and extensibility. Consequently, it can process any workload, for both single-machine products and complex applications. Owned and developed by PostgreSQL Global Development Group, it still remains completely open-source. This DBMS is available for use with such platform systems as Microsoft, iOS, Android, and many more.

Vertical scalability is a hallmark of PostgreSQL, unlike MySQL DBMS. Considering that almost any custom software solution tends to grow, resulting in database extension, this particular option certainly supports business growth and development. PostgreSQL natively supports a large number of data types by default, such as JSON, XML, H-Store, and others. PostgreSQL takes advantage of it, being one of the few relational databases with strong support for NoSQL features. Additionally, it allows users to define their own data types. As your software business model may need different types of databases throughout its existence for better performance or application comprehensiveness, this option brings improved flexibility to the table. Along with that, Postgres is completely open-source and supported by its community, which strengthens it as a complete ecosystem. Additionally, developers can always expect free and prompt community assistance.

On the other hand, while PostgreSQL has a large community and provides strong support for its participants, the documentation still lacks consistency and completeness. As the PostgreSQL community is rather distributed, the documentation doesn’t follow equal standards for all Postgre features. A significant shortcoming of PostgreSQL is the absence of revising tools that would show the current condition of a database. You have to continuously check if something goes wrong. There’s always a risk that DB engineers will notice a failure too late.

**Non-Relational (No-SQL) Databases**

1. **MongoDB**

A free, open-source, non-relational DBMS, MongoDB also includes a commercial version. Although MongoDB wasn’t initially intended for structured data processing, it can be employed for applications that use both structured and unstructured data. In MongoDB, databases are connected to applications via database drivers. They are widely available within the database management system. Multiple types of data are processed simultaneously and use the internal cache for this purpose.

One of the benefits of MongoDB derived from its NoSQL nature is the fast and easy data operation. That is to say, data can be entered, stored, and withdrawn from the database quickly and without any additional confirmation. As with any other non-relational database, it places emphasis on RAM usage, so the records can be manipulated really fast and without any consequences to data integrity. MongoDB is easily combined with different database management systems, both SQL and NoSQL types. Besides that, it has pluggable storage engine APIs, which allows third parties to build their own data storage engines for MongoDB. From a commercial point of view, it creates extra value for business software.  Along with this, scalability; where data is spread out across a distributed network of manageable servers – is a facet of MongoDB’s fundamental nature. It becomes even more important for enterprises operating big data applications. Additionally, the database can allocate data across a cluster of machines. The data is distributed faster and equally, free of bulkiness. As it leads to faster data processing, the application performance is accelerated too.

On the negative side, the denormalization process, when previously normalized data in a database is grouped to increase performance, usually results in high memory consumption. Also, this DBMS keeps in memory all key names for each value pair. Beyond that, because there is no support for joins, Mongo databases have data oversupply, resulting in big memory waste and lower application performance. Also, as MongoDB wasn’t initially developed to deal with relational data models, the performance may slow down in these cases. Besides, the translation of SQL to MongoDB queries takes additional action to use the engine, which may delay the development and deployment.

1. **Redis**

An open-source, NoSQL, in-memory data structure store, Redis can also be used as a cache. Instead of documents, it uses key-value pairs. Its distinct feature is that there are several options for data structuring, such as lists, sets, and hashes. Allowing for data replication and supporting transactions, Redis executes commands in a queue instead of setting it one at a time.

Due to its replication and transaction features, Redis processes the data really fast. The absence of dependencies and in-memory data store type makes Redis a worthy competitor even among simple SQL alternatives. From the data perception and refining perspective, Redis can be considered a colossus. It can easily upload up to 1GB of data for one entry. Add built-in data caching and you get a powerhouse data machine.

On the other hand, total reliance and dependency on the application memory is a real drawback. That is to say, your database will crash if its size exceeds the size of available memory. Regarding compatibility with other dataset types, Redis lags behind. Given that at some time your business may need scaling and using other data formats, having rapid entries as a single option leaves this issue open.

1. **Cassandra**

Cassandra is a free and open-source, distributed, wide-column store, NoSQL database management system designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure. Being rapidly scalable, Cassandra allows for managing large data volumes by replicating them into multiple nodes. It eliminates the problem of database crash – if some of the nodes fail at any time, it’s replaced immediately, and the system keeps working as long as at least one single node is safe.

Due to its master node replication feature, Cassandra stays failure tolerant. It means that DB engineers can feel confident about data safety unless master nodes fail all at the same time. As long as it’s extremely unlikely, the database and the application built on it will stay sound and secure. Cassandra’s simple syntax has the best of SQL and NoSQL. In addition to scalability, it largely contributes to dataset flexibility. Cassandra collects data on the go, and data retrieval shares the same simplicity, despite dataset size. This allows for enlarging the database to the fullest extent.

As Cassandra was initially designed for fast writing, its weakness lies in its incapacity for fast reading. One of the reasons for it is that there are no bottlenecks for information sent, so it needs more time to process. As Cassandra processes multiple layers of data simultaneously, it demands enough power to do it. This means additional investment in both software and hardware. If this is the first time a company faces such a necessity and is not sure about the resources, then maybe it should consider other database systems.

**Conclusion**

there’s a large number of database management systems out there. Each of them is good in its own way, having some drawbacks as well. **If you are just starting a local eCommerce business**, databases like MySQL can be a sensible jumping-off point. **When speaking of analytic tools** without multiple data layers, it may be reasonable to opt for NoSQL databases like MongoDB. It also performs well for product catalogues. Cassandra, is a pretty respectable option for data centers with oceanic volumes of data. Redis is an ideal choice for real-time analytics use cases such as social media analytics, ad targeting, personalization, and IoT.

**References**

**[1] "History of MySQL". MySQL 8.0 Reference Manual. Oracle Corporation. Retrieved 3 April 2020. “MySQL is named after co-founder Monty Widenius's daughter, My.”**

**[2] "Dead database walking: MySQL's creator on why the future belongs to MariaDB - MariaDB, open source, mysql, Oracle". Computerworld. Retrieved 11 September 2013.**