



## Performance Analysis for NoSQL and SQL

Ms. Megha Katkar

ME (Computer Engineering), Shah and Anchor Kutchhi Engineering College, Mumbai, India

---

### ABSTRACT:

With the current emphasis on “Big Data”, NoSQL databases have surged in popularity. These databases are claimed to perform better than SQL database. This trend created a lot of excitement throughout the community of web application developers and among data management developers and researchers. But it also created a lot of debate. Our aim to independently investigate the performance of some NoSQL and SQL databases in the light of key-value stores along with features as Overall time, Transactional integrity and time. A bank application supporting these basic operations is designed and implemented using all the databases tested. Experimental results measure the timing of these operations and we summarize our findings of how the databases stack up against each other. Our results show that NoSQL database with SQL features i.e Foundationdb perform better than SQL and NoSQL databases. And for each database, the performance varies with each operation.

---

**Keywords:**NoSQL, Relational Database, ACID.

---

### I. INTRODUCTION

The infrastructure of a relational database is well-suited to meet the ACID criteria for data. Data is held in tables connected by relational algebra, and transactions are performed in a way that is consistent with ACID principles. But for non-relational databases, such as Bigtable, MongoDB or Dynamo, ACID has always been sacrificed for other qualities, like speed and scalability.

These two classes of systems, relational and NoSQL based systems, represent two opposite points in the scalability versus functionality space. For certain applications, such as web search, email, and social networking, such limited support for transactions in key-value based storage models has been found to be adequate. However, many applications such as online shopping stores, online auction services, financial services, while requiring high scalability and availability, still need certain strong transactional consistency guarantees. For example, an online shopping service may require ACID (atomicity, consistency, isolation, and durability) guarantees for performing payment operations. Thus, providing ACID transactions for NoSQL data storage system is an important problem.

This paper report the comparison between the two leading type of Database storage components prevailing in the industry. The prominent features of both relational as well as non relational databases have been specified which form the basis of the comparison between the two types of database. Our focus is to compare the key-value stores implementations on NoSQL and SQL databases. While NoSQL databases are generally designed for optimized key value stores, SQL databases are not. Yet, our findings suggest that not all NoSQL databases perform better than SQL databases. We compare create, deposit, withdraw and transfer operations related to bank on the key-value storage. We observe that even within NoSQL databases there is a wide variation in the performance of these operations.

### II. DATABASE DESCRIPTION

The key-value category was further divided into in-memory and disk-persistent key-value stores, and the most prominent solutions within each subcategory were chosen. The following are the databases that I have selected for the analysis.

#### A. MySQL:-

MySQL[9] is open source relational database management system. It is more powerful proprietary databases; it has gradually evolved to support higher-scale needs as well. It is still most commonly used in small to medium scale single-server deployments, either as a component in a LAMP-based web application or as a standalone database server. Much of MySQL's appeal originates in its relative simplicity and ease of use, which is enabled by an ecosystem of open source tools such as phpMyAdmin. In the medium range, MySQL can be scaled by deploying it on more powerful hardware, such as a multi-processor server with gigabytes of memory.

Features:-

- Cross-platform support
- Store procedures
- Query caching

- Replication support
- ACID
- Multiple storage engines

#### B. Mongo DB:-

MongoDB[1] (from "HUMONGOUS") is a cross-platform document-oriented database system. Classified as a "NoSQL" database, MongoDB eschews the traditional table-based relational database structure in favor of JSON-like documents with dynamic schemas (MongoDB calls the format BSON), making the integration of data in certain types of applications easier and faster. Released under a combination of the GNU Affero General Public License and the Apache License, MongoDB is free and open source software.

Features:-

- Ad hoc queries
- Indexing
- Replication
- Load balancing
- File storage
- Aggregation

#### C. FoundationDB

FoundationDB[3] is a NoSQL database with a shared nothing architecture. The product is designed around a "core" database, with additional features supplied in "layers."The core database exposes an ordered key-value store with transactions.The transactions are able to read or write multiple keys stored on any machine in the cluster while fully supporting ACID properties. Transactions are used to implement a variety of data models via layers. The FoundationDB Alpha program began in January 2012 and concluded on March 4, 2013 with their public Beta release.

Features:-

- Ordered key-value store
- Transactions
- ACID properties
- Layers
- Commodity clusters
- Replication
- Scalability

#### D.VoltDB

VoltDB [4] is an in-memory relational database that combines Relational, ACID-compliant SQL & the flexibility of JSON, high-velocity data ingestion, massive scalability, and real-time analytics and decisioning to enable organizations to unleash a new generation of applications that act on data at its point of maximum value. VoltDB database can be scaled in two dimensions such as scaling up and scaling out. VoltDB supports a built-in, transactional extract feature and it was designed for High Availability from the ground up.

Features:-

- Automatic partitioning (sharding) across a shared-nothing server cluster
- Memory-centric design
- Automatic replication and disk persistence for high availability
- Data interaction – Relational SQL, JSON data type, JDBC, Ad hoc and stored procedure interfaces
- Integrated Export System for connection to analytic systems
- ACID property

### III. RELATED WORK

While NoSQL databases have the speed and scalability advantage, there have a number of drawbacks compared to traditional relational databases. Leavitt lists these challenges [8]. He notes that NoSQL databases, even though fast for simple tasks, are time-consuming for complex operations. Besides queries for complex operations can be hard to form. The other drawback is the lack of native support for consistency. Leavitt also notes that NoSQL is a technology that many organizations are yet to learn and there is a lack of support and management tools to help. Bartholomew gives a tutorial introduction to the history of and differences between SQL and NoSQL databases [9]. Sakr et al. discuss data management solutions, including NoSQL, for cloud-based platforms [10]. They discuss the challenges data management solutions face in the light of the cloud.

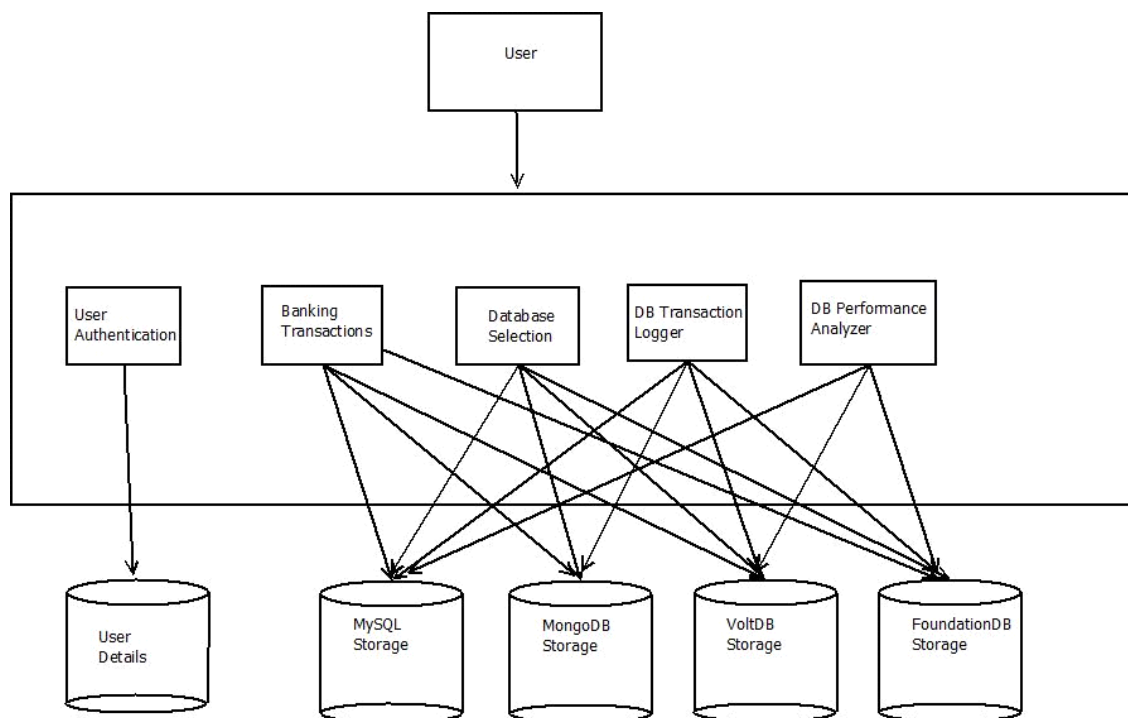
Tiwari provides a comprehensive treatise on NoSQL databases [11]. He covers the history, rationale, programmability, storage architecture, and performance-tuning of some of the implementations.Similarly, Indrawan-Santiago notes the following as the basis of comparison: data model, transaction model, support for ad-hoc queries, indexing, sharding, and license type [12]. She compares ten NoSQL implementations including Cassandra, HBase, Dynamo, MongoDB and CouchDB on this basis. She also qualitatively compares relational databases to NoSQL databases, and concludes that

NoSQL are likely to complement relational databases and enhance an organization's enhance database management capabilities.

Hecht and Jablonski provide a use-case oriented survey of NoSQL databases [13]. They identify the difficulties in choosing a NoSQL database to fit a particular use-case, and therefore focus their paper to address this. They use as the basis for their comparison the data model, support for queries, partitioning, replication, and concurrency controls. They compare in this light fourteen NoSQL databases, including MongoDB, CouchDB, Cassandra and HBase. Boicea et al. compare a NoSQL database against a SQL database. They choose Oracle for the SQL implementation and MongoDB for the NoSQL implementation [14]. They report that, with a large number of records, insertion time is a factor more in Oracle and update and delete times are several factors more in Oracle. Yahoo! Cloud Serving Benchmark is an open-source workload generator tool for comparing key-value stores [15].

#### IV. EXPERIMENTAL FRAMEWORK

The following figure shows the working of system in which user access the application via logging in. He has facilitated to choose the database of his choice whether it is MongoDB,MySQL,VoltDB or FoundationDB.



**Figure 1 : System Block Diagram**

##### A. Workflow –

1. User Authentication - The user is validated here to facilitate valid users to access the system after validation from the backend based on the registered set of users.
2. Banking Transactions - After authentication the user will be given option to demonstrate various kinds of banking transactions like deposit, withdraw and money transfer.
3. Database Transition - Each of the queries generated will be executed on a particular database selected by the user specified during the authentication mechanism.
4. Transaction Logger - Based on the database selected by the user, the system will track the performance achieved by the system in terms of the various parameters
5. Database Performance Analyzer -  
The admin will be facilitated to view the various analyzed parameters in the logger module
  - Disk space usage
  - Time Performance
  - Transactional integrity/consistency
  - CRUD model – Time analysis for various types of queries.

##### B. Database configurations

The database systems can be configured in various ways to take advantage of features such as replication and sharding. For this dissertation the configurations for each database were changed to examine situations and set-ups.

a) Single node

The simplest configuration for databases was to use a single data node without any replicas or shards. VoltDB is slightly more complicated to configure compared to MongoDB, which only requires a single mongod instance to be executed. VoltDB and FoundationDB are implemented using free licence version copy.

b) Database Installation Steps :

MySQL

Download mysql installer(version 5.1.41)

Execute installer

Complete wizard

Check if installation completed successfully by connecting using mysql client at port 3306

Connect using JDBC driver for java using code

MongoDB

Download mongo installer(version 2.6.3)

Execute installer

Complete wizard

Start mongo server using mongod command in command line

Check if server started completed successfully by connecting using mongo client in command prompt

Connect using mongo java client API using code

VoltDB

Download Volt installer for ubuntu(version 4.9)

Execute installer deb file

Check if installation completed successfully by connecting using volt client at default port

Connect using JDBC driver for java using code

FoundationDB

Download Foundation installer(version 2.0.7)

Execute installer (64-bit)

Check if installation completed successfully by connecting using foundation client at default port

Connect using JDBC driver for java using code.

## V. RESULT AND ANALYSIS

For Analysis :

Processing analysis is done in MDB\_Model.java, FDB\_model.java and VDB\_Model.java for three databases in terms of start\_time and end\_time variables and the difference between them for identifying the final values.

And for mysql the corresponding servlets contain the data required for creating analysis log data. like deposit.java, withdraw.java, Transfer.java.

A. Overall Time Performance:

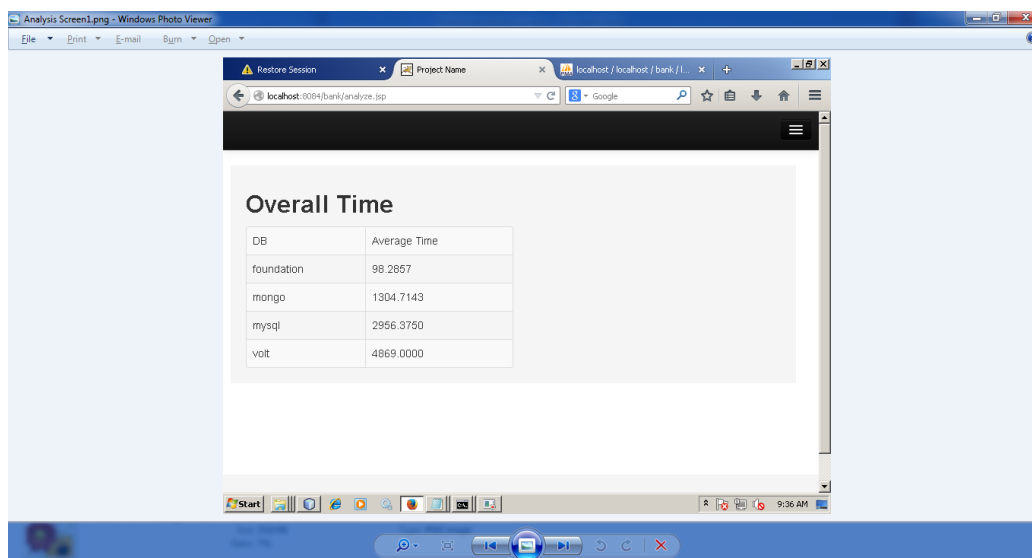
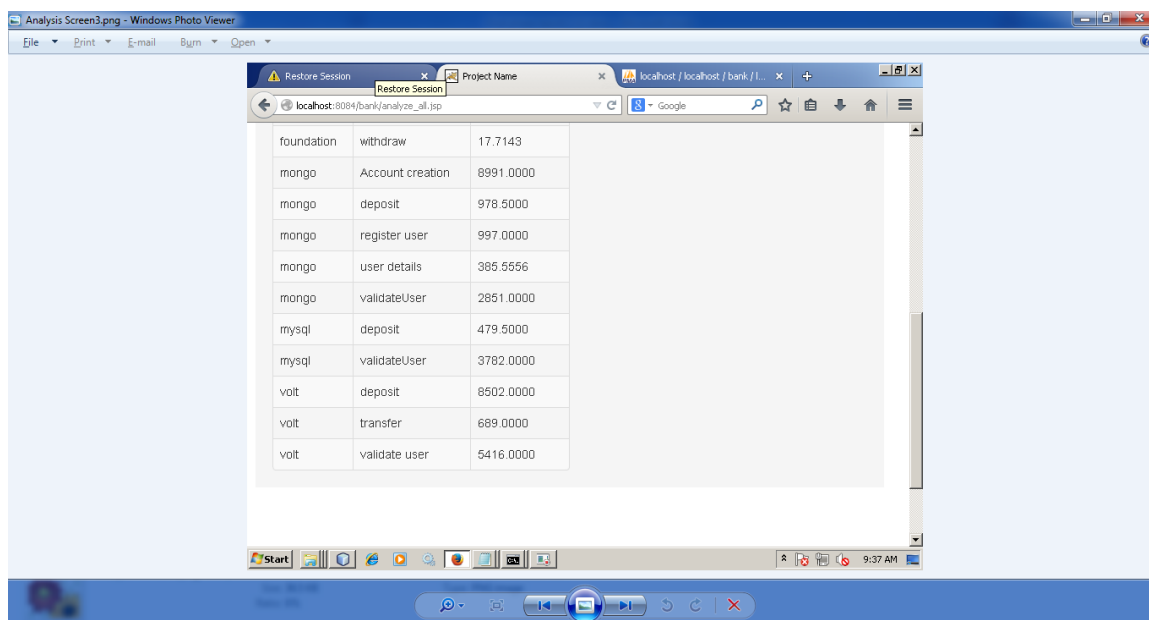


Figure 2: Overall Time Performance

Report Interpretation-

- This Report shows the overall time taken by each database to complete the transactions.
- This report has obtained from calculating the overall time taken by databases to execute each transaction.

B. Transactionwise performance of each database:



| Database   | Transaction      | Time (ms) |
|------------|------------------|-----------|
| foundation | withdraw         | 17.7143   |
| mongo      | Account creation | 8991.0000 |
| mongo      | deposit          | 978.5000  |
| mongo      | register user    | 997.0000  |
| mongo      | user details     | 385.5556  |
| mongo      | validateUser     | 2851.0000 |
| mysql      | deposit          | 479.5000  |
| mysql      | validateUser     | 3782.0000 |
| voit       | deposit          | 8502.0000 |
| voit       | transfer         | 689.0000  |
| voit       | validate user    | 5416.0000 |

**Figure 3: Transaction wise Performance of each database**

Report Interpretation-

- This Report shows the time taken by each database to complete various transactions.
- This report shows the variation of time for each transaction for various databases.

Report Analysis:

- The time taken by foundationdb to perform the transactions is comparatively less than other three databases.
- While comparing between the MySQL and VOITDB's performance, the time taken by each database to perform same transaction varies.

## VI. CONCLUSIONS

Our proposed system allow user to select the database to perform his transactions. On the basis of that system keeps track of each transaction performed by different users on different databases via transaction logger. Using this logger to compare the databases, we find that the FoundationDB which is a NoSQL database with SQL layer performs better than SQL and NoSQL databases. As compared with VoltDB, Overall time taken by FoundationDB is less.

## REFERENCES

- [1] C. Strauch, "NoSQL Databases," February 2011. [Online]. Available: <http://www.christofstrauch.de/nosql dbs.pdf>.
- [2] P. Warden, Big Data Glossary. O'Reilly Media, September 2011.
- [3] <https://foundationdb.com>
- [4] [voltdb.com](http://voltdb.com)
- [5] "Managing Big Data:SQL or NoSQL?" by Carl W. Olofson Filing Information: January 2012, IDC #232706, Volume: 1, Tab: Vendors Big Data: Global Overview: Technology Assessment
- [6] J. Han, E. Haihong, G. Le, and J. Du, "Survey on NoSQL database," in Pervasive Computing and Applications (ICPCA), 2011 6th International Conference on, oct. 2011, pp. 363 –366.
- [7] K. Chodorow and M. Dirolf, MongoDB: The Definitive Guide. O'Reilly Media, September 2010.
- [8] N. Leavitt, "Will NoSQL databases live up to their promise?" Computer, vol. 43, no. 2, pp.12 –14, feb. 2010.
- [9] D. Bartholomew, "SQL vs. NoSQL," Linux Journal, no. 195, July 2010.
- [10] S. Sakr, A. Liu, D. Batista, and M. Alomari, "A survey of large scale data management approaches in cloud environments," Communications Surveys Tutorials, IEEE, vol. 13, no. 3, pp. 311–336, 2011.

- [11] S. Tiwari, Professional NoSQL. Wiley/Wrox, August 2011.
- [12] M. Indrawan-Santiago, "Database research: Are we at a crossroad? Reflection on NoSQL," in Network-Based Information Systems (NBIS), 2012 15th International Conference on, sept. 2012, pp. 45 –51.
- [13] R. Hecht and S. Jablonski, "NoSQL evaluation: A use case oriented survey," in Cloud and Service Computing (CSC), 2011 International Conference on, dec. 2011, pp. 336 –341.
- [14] A. Boicea, F. Radulescu, and L. I. Agapin, "MongoDB vs Oracle – database comparison," in Emerging Intelligent Data and Web Technologies (EIDWT), 2012 Third International Conference on, sept. 2012, pp. 330 –335
- [15] B. F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, and R. Sears, "Benchmarking cloud serving systems with yesb," in Proceedings of the 1st ACM symposium on Cloud computing, ser. SoCC '10. ACM, 2010, pp. 143–154.