Performance Comparison between Key-Value NoSQL Databases

Preeti Bhatt*
** D.C.S.A. Panjab University sector-14 Chandigarh*
Preeti.gcg@gmail.com

Abstract- These days, large number of applications produces unstructured content which is very complex or impossible to manage with relational databases. These applications support cloud computing and demands a database that can handle these unstructured data as well as structured data without lacking scalability, consistency and availability. Key-value store databases are not the complete replacement of relational databases but help in handling unstructured, semi-structured and structured data. In this paper we describe Key-value store database, their features and perform comparison between various key-value store databases.

Keywords- Non-Relational, NoSQL, Key-value Store Databases, Scalability, Distributive Databases.

I.INTRODUCTION

A key-value database is a type of NoSQL database that stores data in the form of a simple key/value pair. Many programming languages uses this concept of key-value pair typically referred as an associative array or data structure. Hash or data dictionary is also example of key-value.

Table -1 Phone Directory

Key	**Value**
Ramesh	(123) 456-7890
Saurabh	(234) 567-8901
Tina	(345) 678-9012
Tarun	(456) 789-0123

1.1 What Type of Data can be stored in a Key-Value Database?

The Key: In the key-value pair, key must be unique that allow to access the value associated with that key. Key size and data type may vary in different NoSQL databases. For example, In Redis, key size of at most 512MB is allowed. Binary sequence, image file, short string can be used as a key. Key should not be too long and not too short. A key that is too long should be avoided. A key that is too short cause readability issues.

The Value: Key value can be a number, short text or long text, programming code such as VBScript, JavaScript, PHP, HTML markup code, an image, list or an object which encapsulate another key-value pair etc.

1.2 When to use key-value stores?

Key-value store database can be used to store following:- Emails, User profiles, blog comments, Ecommerce, Session information, Product categories, Product reviews, Data deduplication, Telecom directories, Product details, Status messages, Shopping cart contents, Internet Protocol (IP) forwarding tables, Networking data maintenance, Product recommendations, Session management at high scale. Example of Key-Value store databases are:- Redis [16], Oracle NoSQL Database, Voldemorte, Aerospike [18], Oracle Berkeley DB, DynamoDB [14], Couchbase [15], Riak [17], memcached [19].

1.3 Features of key-value stores database?

- Flexible data structure (schema-free)
- There is no need of schema definition or upfront data modeling because the value is stored in blob.
- Blob removes the need to index the data to improve performance.
- Closely follows object-oriented programming concepts.

- Key-value uses less memory to store database lead to large performance gain.
- Key value store databases uses no query language. Get, put and delete commands are used to store, retrieve and update data.
- Simple data retrieval by direct request on disk or to the object in memory makes key-value store databases fast, scalable, easy to use, flexible, and portable.

II.RELATED WORK

Amazon developed storage technologies such as Amazon S3 [1] to achieve reliability and scaling; uses a synthesis of various technologies to meet scalability and availability needs. We studied the literature [2] in which the author analyzed the performance of CouchDB and Elasticsearch based on insertion, deletion, updating and selection operations and conclude that CouchDB works efficiently on insertion, deletion and update operation but Elasticsearch works much better in case of selection operation. [3] Did performance comparison between MongoDB, Redis, couchbase, Cassandra and HBase. [4,5] describe a lab-based benchmark which uses a measurement tool named YCSB(Yahoo Cloud Serving Benchmark).Some features in NoSQL databases are similar as Relational database but the behaviors are different. [6] Did some performance test on Hbase column-store database, including column-family test conclude that when we increase number of column families then the reading and writing speed becomes slower because in Hbase each column family is stored separately. [7] Suggests Yahoo! Cloud Serving Benchmark(YCSB) for key-value store database as its able to perform read, write, update, scan operation well. [8] Describe the architecture of Couchbase servers and its evolution in detail.
Couchbase internal architecture supports OLTP like workload. [9] Compares and tests NoSQL databases based on main characteristics such as data loading, reads, only read, read –modify-write, only updates and updates using YCSB as measurement tool. [10] Compares and tests MongoDB and Cassandra based of main characteristics such as data loading, only read, read –modify-write only updates, reads and updates using YCSB as measurement tool. [11] analyzed the Memchached database against size of key-value pair and number of client and perform optimization techniques such as Interrupt blanking and zero copying.[12] introduced a high performance key-value benchmark that can interconnect web-scale workloads. [13] Proposed the client-coordinated transaction protocol that is built as a Java library that gives multi item transaction over heterogeneous key value data stores and YCSB benchmark is used as measurement tool.

III.COMPARITIVE ANALYSIS OF KEY-VALUE STORE NOSQL DATABASES

Table1: The Comparison and contrast of Key-Value NoSQL databases

	Redis	**DynamoDB**	**Aerospike**	**Couchbase**	**Memchached**
Development language	C	Java, Net, Perl, JavaScript, C#	C	C, C++, Erlang, Go	C
Storage type	Key-value store	Document store Key-value store	Key-value store	Key-value store Document store	Key-value store
Developer	Salvatore Sanfilippo (developer at Redis Lab) (2009)	Amazon (2012)	Aerospike (2012)	Couchbase, Inc. (2011)	Danga Interactive (2003)
Transactions	Optimistic locking, atomic execution of commands blocks and scripts	ACID	Atomic execution of operations	Single-document ACID transactions	No
Replication	Master-slave replication Multi-master replication	Yes	selectable replication factor	Master-master replication Master-slave replication	No
Concurrency	Not Supported	Optimistic Concurrency Control (OCC)	Yes	Optimistic Concurrency Control (OCC)	Not Supported
Triggers	No	Yes	No	Yes	No
CAP Theorem	Consistency, Partition tolerance	High Availability, Partition tolerance	both AP(Availability and Partition tolerance) mode	Consistency, Partition tolerance	Consistency, Partition tolerance

			and Strong Consistency Mode		
Operating System/Platform	POSIX Systems	Cross-platform	Linux / Unix-like	Cross platform	Cross platform
High Availability	No	Yes	Yes	No	No
License type	Open Source (BSD 3-clause)	Commercial	Open Source AGPL(Affero General Public License)	Open Source Apache License 2.0	Open source (BSD license)
Map reduce	No	No	Yes	Yes	No
Consistency	Strong eventual consistency with CRDTs Eventual Consistency	Eventual Consistency, Immediate Consistency	Eventual Consistency in cross-datacenter configuration and Immediate Consistency in local cluster configuration	Eventual Consistency Immediate Consistency	Yes
Querying	Custom API Complex query support	SQL	AQL	Declarative query language (N1QL) that extends ANSI SQL to JSON. First commercial implementation of SQL++.	API Calls Set, Add, Get, Delete
Partitioning scheme	Consistent Hashing, Sharding	Sharding	Sharding	Sharding	None
Scalability	Liner scalability	Incremental (Massive and seamless scalability)	linear scalability	Multi-dimentioal scalability	High Scalability
API	RESP-Redis Serializable Protocol	RESTful HTTP API (Put, get)	Proprietary protocol JDBC	Native language bindings for CRUD, Query, Search and Analytics APIs	Custom API Get,Set operations
Storage Architecture	Disk-oriented	Disk-oriented	Disk-oriented	Disk-oriented	In-Memory
System isolation	Serializable	Read, Uncommitted Read, Committed Repeatable Read	Multi-record isolation and consistency	Read Committed, Serializable	No
Data Scheme	Scheme free	Scheme free	Scheme free	schema-free	schema-free
Foreign keys	Not Supported	Not Supported	Not Supported	Not Supported	Not Supported
Joins	Not Supported	Not Supported	Not Supported	Nested Loop Join	Not Supported
Stored Procedures	Not Supported	Not Supported	Not Supported	"Prepared queries" supported	Not Supported
Views	Not Supported	Not Supported	Not Supported	Materialized Views	Not Supported
Logging	Command Logging	Not Supported	Log callback interface	Logical Logging	Not Supported
Indexes	Hash Table	Not Supported	Hash Table with a distributed tree structure	B+Tree Skip List Hash Table Inverted Index (Full Text)	Hash Table
Cloud-based only	No	Yes	No	No	No
User Concept	Simple password-based access control	Access rights for users and roles can be defined via the AWS Identity and Access Management (IAM)	Access rights for users and roles	User and Administrator separation with password-based and LDAP integrated Authentication	Uses SASL(Simple Authentication and Security layer) Protocol
Durability	Yes	Yes	Yes	Yes	No
Server-side scripts	Lua	No	user defined	Functions and	No

			functions with Lua	timers in JavaScript	

IV.CONCLUSION

In this paper we describe key-value store databases, their features and perform comparative analysis of various key-value store databases. We conclude that the Redis database is rich in features and widely popular key-store database. Memcached is used as a caching system. Although Redis and Memcached doesn't provide concurrency and availability but provide strong consistency. Memcached has in-memory storage architecture but not durable and don't have isolation property because it doesn't support transactions and concurrency.

REFERENCES

[1] DeCandia G, Hastorun D, Jampani M, Kakulapati G, Lakshman A, Pilchin A, Sivasubramanian S, Vosshall P, Vogels W. Dynamo: amazon's highly available key-value store. InACM SIGOPS operating systems review 2007 Oct 14 (Vol. 41, No. 6, pp. 205-220). ACM.

[2] Gupta S, Rani R. A comparative study of elasticsearch and CouchDB document oriented databases. In2016 International Conference on Inventive Computation Technologies (ICICT) 2016 Aug 26 (Vol. 1, pp. 1-4). IEEE.

[3] Tang E, Fan Y. Performance comparison between five NoSQL databases. In2016 7th International Conference on Cloud Computing and Big Data (CCBD) 2016 Nov 16 (pp. 105-109). IEEE

[4] Tudorica BG, Bucur C. A comparison between several NoSQL databases with comments and notes. In2011 RoEduNet international conference 10th edition: Networking in education and research 2011 Jun 23 (pp. 1-5). IEEE.

[5] Cooper BF, Silberstein A, Tam E, Ramakrishnan R, Sears R. Benchmarking cloud serving systems with YCSB. InProceedings of the 1st ACM symposium on Cloud computing 2010 Jun 10 (pp. 143-154). ACM.

[6] Naheman W, Wei J. Review of NoSQL databases and performance testing on HBase. InProceedings 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC) 2013 Dec 20 (pp. 2304-2309). IEEE.

[7] Patil S, Polte M, Ren K, Tantisiriroj W, Xiao L, López J, Gibson G, Fuchs A, Rinaldi B. YCSB++: benchmarking and performance debugging advanced features in scalable table stores. InProceedings of the 2nd ACM Symposium on Cloud Computing 2011 Oct 26 (p. 9). ACM.

[8] Borkar D, Mayuram R, Sangudi G, Carey M. Have your data and query it too: From key-value caching to big data management. InProceedings of the 2016 International Conference on Management of Data 2016 Jun 26 (pp. 239-251). ACM.

[9] Jayathilake D, Sooriaarachchi C, Gunawardena T, Kulasuriya B, Dayaratne T. A study into the capabilities of NoSQL databases in handling a highly heterogeneous tree. In2012 IEEE 6th International Conference on Information and Automation for Sustainability 2012 Sep 27 (pp. 106-111). IEEE.

[10] Abramova V, Bernardino J. NoSQL databases: MongoDB vs cassandra. InProceedings of the international C* conference on computer science and software engineering 2013 Jul 10 (pp. 14-22). ACM.

[11] Chidambaram V, Ramamurthi D. Performance analysis of memcached. unpublished Manuscript.[Online]. Available: http://citeseerx. ist. psu. edu/viewdoc/download.

[12] Shankar D, Lu X, Wasi-ur-Rahman M, Islam N, Panda DK. Benchmarking key-value stores on high-performance storage and interconnects for web-scale workloads. In2015 IEEE International Conference on Big Data (Big Data) 2015 Oct 1 (pp. 539-544). IEEE.

[13] Dey A, Fekete A, Röhm U. Scalable transactions across heterogeneous NoSQL key-value data stores. Proceedings of the VLDB Endowment. 2013 Aug 28;6(12):1434-9.

[14] https://aws.amazon.com/dynamodb/

[15] https://www.couchbase.com/

[16] https://redis.io/

[17] https://riak.com/

[18] https://www.aerospike.com/

[19] https://memcached.org/