iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 6 **Issue:** VI **Month of publication:** June 2018

DOI: http://doi.org/10.22214/ijraset.2018.6184

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue VI, June 2018- Available at www.ijraset.com

Performance Evaluation of Redis and MongoDB Databases for Handling Semi-structured Data

Gurpreet Kaur Spal[1], Prof. Jatinder Kaur[2]
[1,2]*Department of Computer Science and Engineering, Baba Banda Singh Bahadur Engineering College, Fategarh Sahib*

***Abstract**: Today, the information is growing rapidly by online networking platforms. The need for high-speed, distributed, manageability, open-source leads to introducing NoSQL databases. Therefore, relational database may not good with the developing necessities that need more accomplished system. So, choose Redis and MongoDB databases from NoSQL family because they are robust open source databases. But still needs to investigate which NoSQL database is best in reading and writing operations and check the performance from processing time. The relational database does not support to store data in variety of formats because it has inflexible storage structure like table in database. These limitations are only tackling with NoSQL databases and offer preferred storage option for semi-structured data. The results have shown that Redis database has good runtime performance as compared to MongoDB database in performing read and write operations.*
***Keywords**: NoSQl database, Redis, MongoDB, Key-value store, Document-oriented store.*

I. INTRODUCTION

The origin of information is generated by online networking platforms and is growing with massive speed. The complexity emerge from the speed of information generation and the requirement for brief time or continuous information handling, from heterogeneous information sources, from semi-organized or unstructured information things, and from managing unfinished information because of external elements[1]. In spite of the fact that the relational database is well known, it may not be good with the developing necessities that require more accomplished system. The relational database is incompatible to store data in variety of formats because of an inflexible storage structure. The difficulty is that to read and write data at a much faster rate.

A better way is to look for any open source database which is freely available and look from NoSQL databases family. The previously mentioned limitations are tackled with NoSQL databases and provide preferred storage option for semi-structured data. Therefore, non-relational databases have no rigid database schema, provide more flexibility and can store information in variety of formats. NoSQL, which remain for “not only SQL” is a contrasting option to relational databases in which information is put in tables. NoSQL databases are particularly valuable for working with huge arrangements of appropriated size and to exhibit the data in a more suitable form [2]. They choose Redis and MongoDB as NoSQL databases because both are robust open source databases.

In this paper, the aim is to compare the performance between Redis (key-value store) and MongoDB (document-oriented store) as NoSQL databases. They investigate which NoSQL database is better in read and write operations and measure the performance from processing time. At last the experimental results shows that Redis database is best in reading and writing operations as compared to MongoDB database.

II. NOSQL DATA MODELS

In this section, NoSQL databases are organize into four different data models. Scale horizontally and distributed are more basic attributes of all NoSQL databases. There are numerous NoSQL databases present, but, they constitute different data models.

A. Key-value Stores

In this data model, the data is stored into two sections as key-value pair where the string that display the key and the real information that displays the value. The data model structure is extremely simple, effective and query performance is much better than traditional relational database. A concept of this model is equivalent to hash tables and resulting in values being indexed by keys for information retrieval. It supports mass storage, handle structured or unstructured information and provide consistency. The most general advantage of key-value store is to perform read/write operation in less time as compared to relational database[3]. Some examples of key-value databases are Redis, Scalaris, Riak and so on.

B. Document-oriented Stores

Basically, this database contains data model equivalent to key-value store and put complex information in document form (for example JSON, XML or PDF documents). A document-oriented store can contain a number of documents called collection, the data

is put inside documents where documents contain key-array pairs, or key-value pairs, or nested documents. Documents are retrieved or find by using a unique key which might be basic string. It provides high performance, horizontal scalability and do not have any schema restrictions[4]. Some examples of document-oriented databases are MongoDB, CouchDB, SimpleDB and so on.

C. Column-oriented Stores

data model of column-oriented database can store data in columns, data tables where each column can store data separately and indexed in database. The basic structure of column-family store is made up of row key (i.e., a unique identifier) and column (contains a name, a value, and timestamp). It can identify the most recent version of information by timestamp. The column-family stores have schema flexibility, high scalability, support complex modelling structures like repeating groups, set, list, nested tables, aggregation, etc. The aggregation queries of column-oriented databases are SUM, COUNT, AVG, etc. Moreover, this data model is very scalable and support clustering for spreading data over a large number of machines. It can load data at extremely high speed[5]. Some examples of column-oriented databases are Cassandra, HBase, HyperTable and so on.

D. Graph Stores

The data model of graph databases is based on graph theory that emphasis on relationship between information. The structure of graph stores is made up of nodes (i.e., an object or an entity in the database), edges (i.e., the relationship between the objects) and properties. Using graph based NoSQL database, information can more simply transfer from one model to another. The graph model approach is simple and easier for development, documentation and sharing of information to other models. The complicated hierarchical structures of information become simple after designed by graph theory. It can provide fast retrieval of information. This data model is horizontally scalable and flexible because it can be used over multiple machines. The application areas of graph databases are location based services, knowledge presentation and path discovering issues, suggestion systems and so on[4]. Some examples of graph databases are Neo4j, OrientDB and so on.

III.EXPERIMENTAL DESIGN

A. Read and Write Sample Data in Redis Database

This section illustrates the detail of “Read and Write sample data in Redis Database” and also discusses the implementation steps in the flow chart. It mainly considers the method which improves the system performance by reading and writing data more frequently and also saves time.

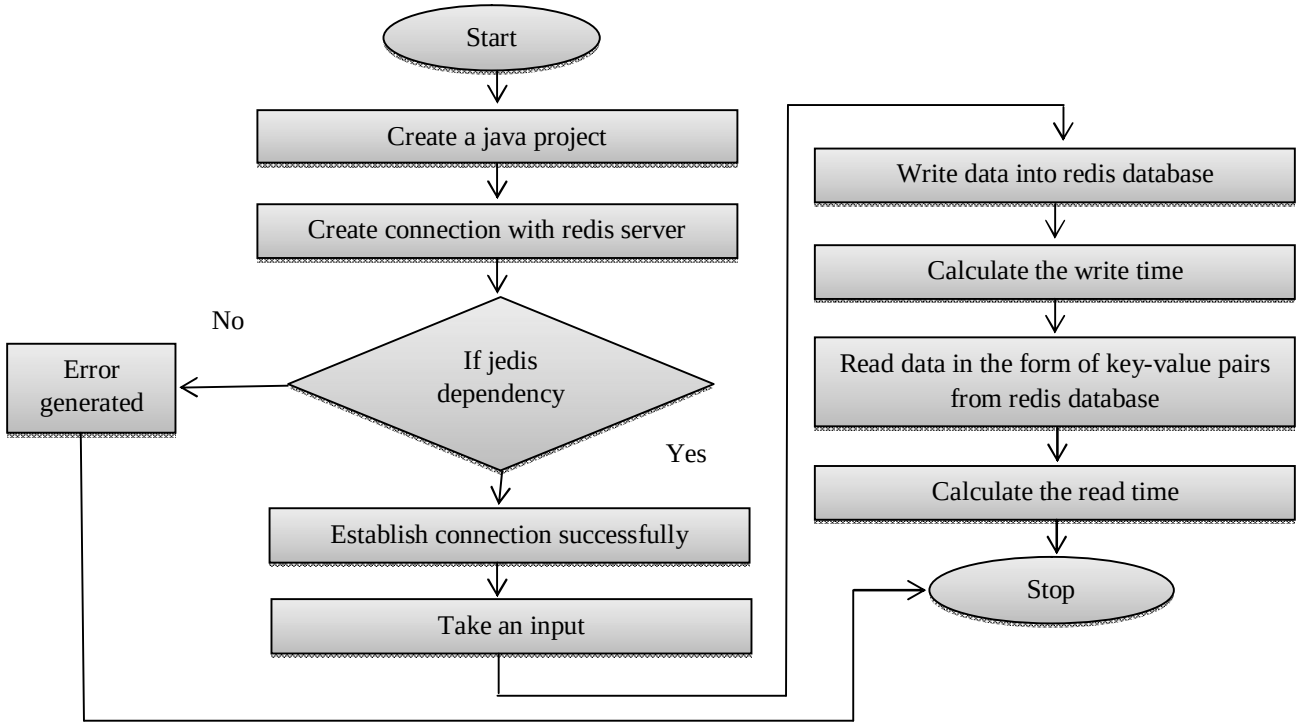


Fig. 1 Flow chart of read and writes sample data in redis database

Step 1: Start
Step 2: Create a java project.
Step 3: Create connection with redis server.
Step 4: If there is no jedis dependency then generates error.
Step 5: By adding jedis dependency that helps in making connection successful with redis database.
Step 6: Take an input as CSV (Comma Separated Values) data.
Step 7: Write data into redis database using SET command.
Step 8: Calculate the insertion time of key-value pairs in redis database.

B. Read and Write Data in MongoDB Database

In this subsection, illustrate the detail about the experimental design of "Read and Write sample data in Mongodb database" and also discuss the implementation steps of the following design in the flow chart.

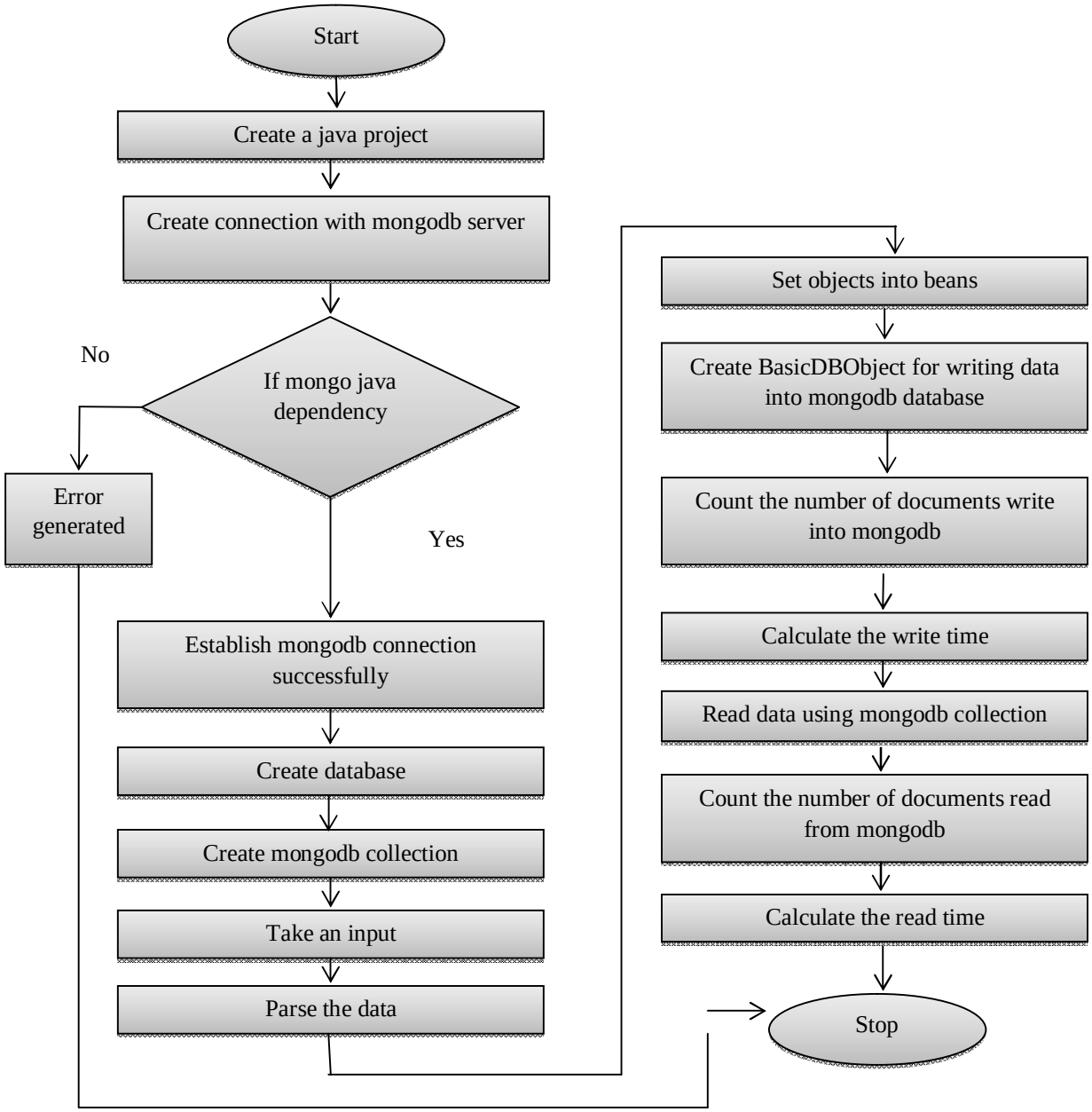


Fig. 2 Flow chart of read and writes data in mongodb database

Step 1: Start
Step 2: Create a java project.
Step 3: Create connection with mongodb server.
Step 4: If there is no mongo java dependency then generates error.
Step 5: By adding mongo java dependency that helps in making connection successful with mongodb database.
Step 6: Take an input as CSV (Comma Separated Values) data.
Step 7: Parse the input data for process the text line into words (tokens).
Step 8: Set objects into beans by matching entity of their attributes.
Step 9: Create BasicDBObject for making documents in database and it helps in inserting data in fields and value pairs.
Step 10: Count the number of documents writing into mongodb database.
Step 11: Calculate the writing time.
Step 12: Read data using mongodb collection.
Step 13: Count the number of documents reading from the mongodb database.
Step 14: Calculate the read time.
Step 15: Stop.

IV. EXPERIMENTAL SETUP

For this experimental setup, need to install both Redis and MongoDB on the same machine that is running an AMD A8-7410 APU processor operating at 2.20 GHz with 2 GB or above of memory. The experiment is developed on Windows operating system and is coded in Java and using Redis and MongoDB server. For Redis, the use of a library which named jedis for connection between java framework and Redis database and used the version 2.8.1. The use of Mongo java driver for establishes connection between java framework and MongoDB database and the use of latest version 3.6.3.

V. EXPERIMENTAL RESULTS

A. Compare the Writing time of Redis and MongoDB Databases

Write operations have been done in both databases with different sizes of data are 2kb, 11kb, 34kb, 2mb, 11mb and 35mb respectively. In MongoDB, inserted the files with different sizes into collection in the form of documents and have measured the total time taken to insert all the data files and compare the results in this section. To write data, only one collection in MongoDB named dummy have been used. The results show that MongoDB performs INSERT operations with not higher speed than Redis database in all six different sizes of data, and with the increase in number of data size the results become more clear. The results are shown in Fig. 3.

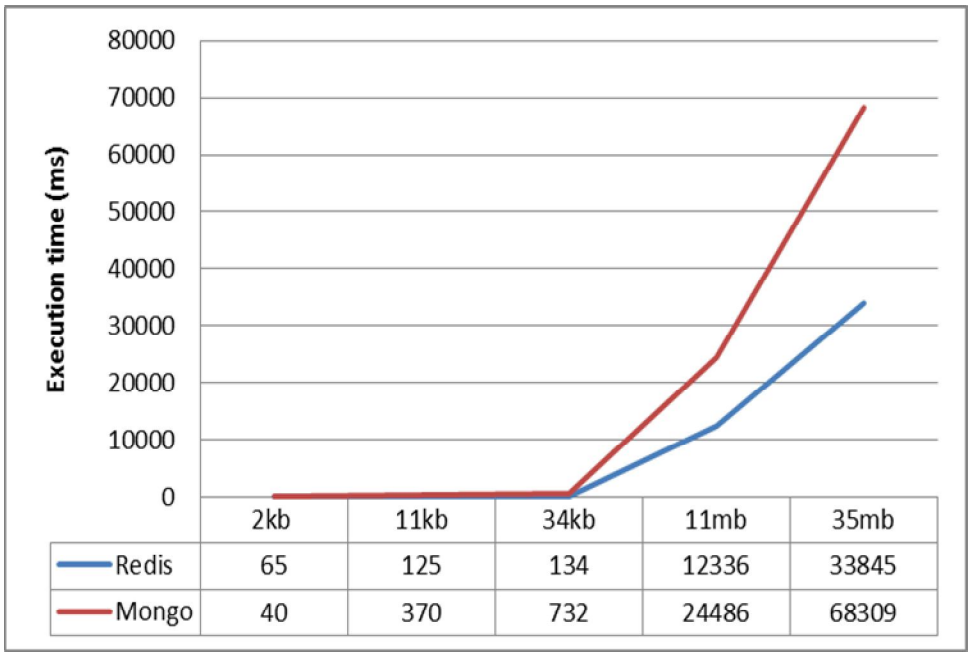
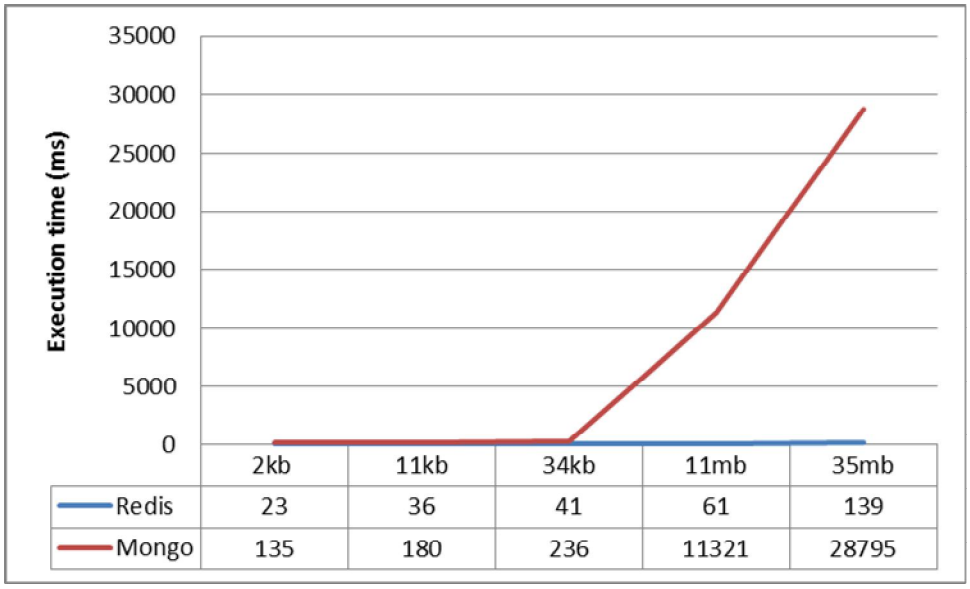


Fig. 3 Comparisons of write time

B. Compare the Reading time of Redis and MongoDB Databases

In this subsection, to perform read operations with the six different sizes of data that mentioned in the previous subsection. To read data, only one collection in MongoDB named dummy have been used. As can be seen from the results, Redis database engine has absolute superiority. The results are shown in Fig. 4.



	2kb	11kb	34kb	11mb	35mb
Redis	23	36	41	61	139
Mongo	135	180	236	11321	28795

Fig. 4 Comparisons of read time

VI. CONCLUSIONS

This work concludes the importance of NoSQL databases in processing vast amount of data at higher rate as compared to relational database management system. It is defined that the traditional relational database was not to be much effective thus NoSQL databases are used in this work to overcome the shortcomings of previous one. Therefore, it is very important to look at the features and operations of various NoSQL databases to decide which database is suitable for big data applications. On the basis of the results, it is observed that the Redis database is blazing fast in performing read and write operations in comparison to MongoDB database.

In future, plan to evaluate the application of Redis and MongoDB databases in a distributed environment and analyse the execution time with the increase in the number of operations to be performed.

REFERENCES

[1] Bohlouli, M., Schulz, F., Angelis, L., Pahor, D., Brandic, I., Atlan, D. and Tate, R., 2013. Towards an integrated platform for big data analysis. In Integration of practice-oriented knowledge technology: Trends and prospectives (pp. 47-56). Springer, Berlin, Heidelberg.

[2] Puangsaijai, W. and Puntheeranurak, S., 2017, March. A comparative study of relational database and key-value database for big data applications. In Electrical Engineering Congress (iEECON), 2017 International (pp. 1-4). IEEE.

[3] Sitalakshmi Venkatraman, K.F., Kaspi, S. and Venkatraman, R., 2016. SQL Versus NoSQL Movement with Big Data Analytics. IJ Information Technology and Computer Science Information Technology and Computer Science, 12(12), pp.59-66.

[4] Makris, A., Tserpes, K., Andronikou, V. and Anagnostopoulos, D., 2016. A classification of NoSQL data stores based on key design characteristics. Procedia Computer Science, 97, pp.94-103.

[5] Storey, V.C. and Song, I.Y., 2017. Big data technologies and Management: What conceptual modeling can do. Data & Knowledge Engineering, 108, pp.50-67.

[6] Banker, K., 2011. MongoDB in action. Manning Publications Co..

[7] Wei-Ping, Z., Ming-Xin, L.I. and Huan, C., 2011, May. Using MongoDB to implement textbook management system instead of MySQL. In Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on (pp. 303-305). IEEE.

[8] Carlson, J.L., 2013. Redis in Action. Manning Publications Co..

[9] Lotfy, A.E., Saleh, A.I., El-Ghareeb, H.A. and Ali, H.A., 2016. A middle layer solution to support ACID properties for NoSQL databases. Journal of King Saud University-Computer and Information Sciences, 28(1), pp.133-145.

[10] El Alami, A. and Bahaj, M., 2016, September. Migration of a relational databases to NoSQL: The way forward. In Multimedia Computing and Systems (ICMCS), 2016 5th International Conference on (pp. 18-23). IEEE.

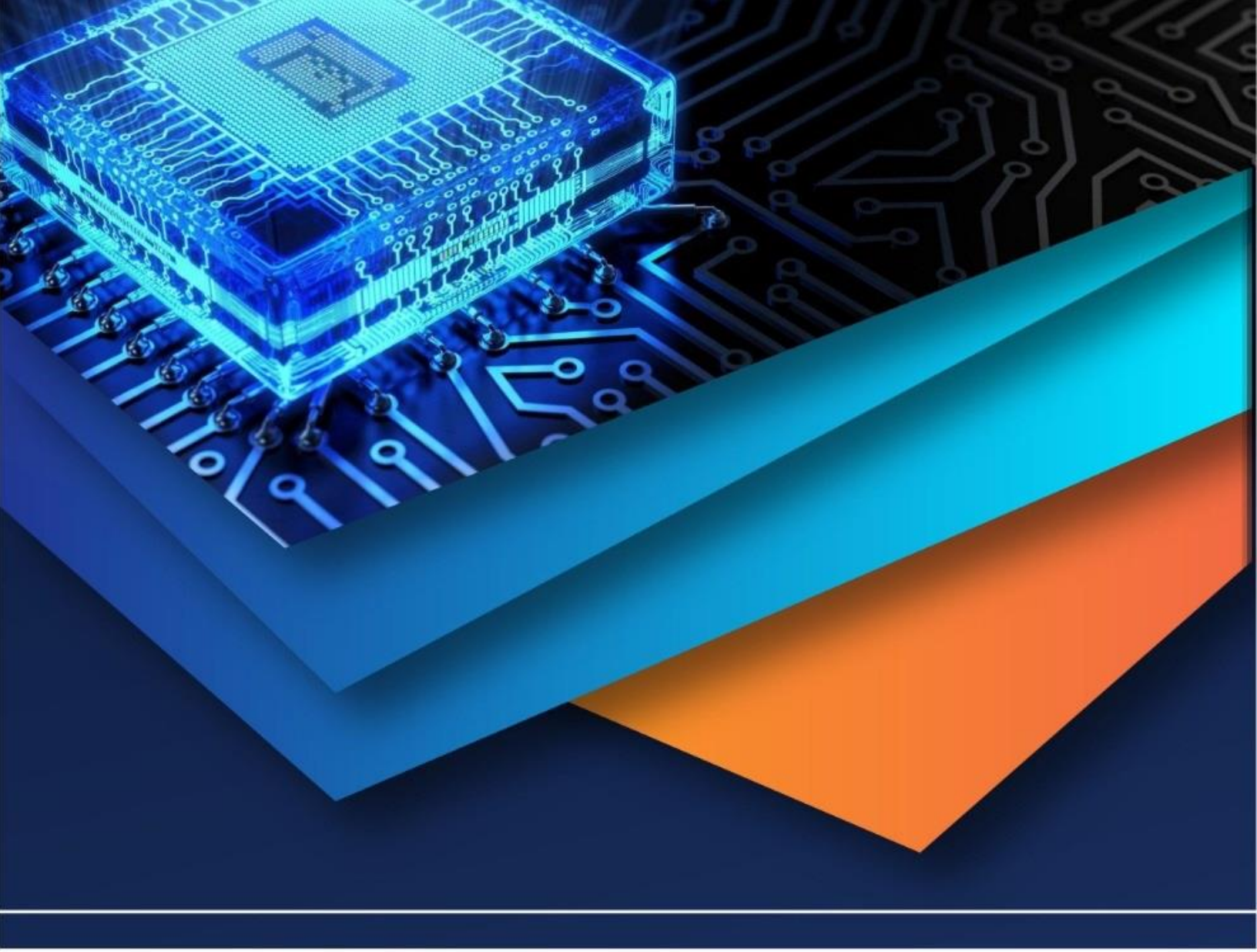
[11] Guler, B. and Ozkasap, O., 2017, July. Compressed incremental checkpointing for efficient replicated key-value stores. In Computers and Communications (ISCC), 2017 IEEE Symposium on (pp. 76-81). IEEE.

[12] Lawrence, R., 2014, March. Integration and virtualization of relational SQL and NoSQL systems including MySQL and MongoDB. In Computational Science and Computational Intelligence (CSCI), 2014 International Conference on (Vol. 1, pp. 285-290). IEEE.

[13] Panda, R. and John, L.K., 2017, September. Proxy benchmarks for emerging big-data workloads. In Parallel Architectures and Compilation Techniques (PACT), 2017 26th International Conference on (pp. 105-116). IEEE.

[14] Bazar, C. and Iosif, C.S., 2014. The transition from rdbms to nosql. a comparative analysis of three popular non-relational solutions: Cassandra, mongodb and couchbase. Database Systems Journal, 5(2), pp.49-59.

[15] Zaki, A.K. and Indiramma, M., 2015, March. A novel redis security extension for NoSQL database using authentication and encryption. In Electrical, Computer and Communication Technologies (ICECCT), 2015 IEEE International Conference on (pp. 1-6). IEEE.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY