Proc. of Int. Conf. on Current Trends in Eng., Science and Technology, ICCTEST

Storage Implications On IOT Data Using NOSQL In Cloud

Shruthi.G [1]and Veerendra Patil H.V [2]
[1]Asst.Professor, Dept.of CSE,DBIT, Bangalore, India
[2]Asst.Professor, Dept.of CSE,DBIT, Bangalore, India

***Abstract*— The Internet of Things (IoT) is an important topic in Information technology, industry and engineering circles that are expected to change the next era of computing. The convergence between cloud computing and IoT has become important topic over the past years because of the benefits that IoT. The Internet of Things (IoT) remains a main topic for standardization and attracts interest from industry, public authorities and end users.**

***Index Terms*— ICT; storage; challenges; datacenter; language; query, Networks.**

I. INTRODUCTION

The state of the internet of things has developed due to a exchange of information of multiple technologies, ranging from wireless to the Internet and from embedded systems to micro-electromechanical systems. The traditional fields of embedded systems, wireless sensor networks, system, automation and others all contribute to enabling the internet of things this kind of a network brings a lot of challenges for data storage and processing in a cloud platform. IoT data can be generated at a greater rate , the amount of data can be huge and the types of data can be various. In order to address these future problems, in this paper proposes a databases issues in IOT and also comparative study of NOSQL database.

II. THE CLOUD-BASED INTERNET OF THINGS

In Cloud computing, most of the computing resources exist on the Internet on servers, as opposed to client machines such as laptops or personal computers. Cloud computing is commonly associated with Information Technology services, but can theoretically be extended to embedded software programming [10]. Integrating Cloud computing with Wireless Sensor Networks (WSNs) brings the concept of Cloud-based embedded system programming. The Cloud-based integrated programming environment has a common benefit, namely that the local administrators and users don't need to spend time with large client and server machine installations, setups, or software updates. With Cloud-based tools, the user can program from anywhere that has an Internet connection. In the proposed Cloud-based model [10], the Cloud connects devices such as PCs, smart phones, embedded development platforms, or host machines to Cloud-based programming tools. These tools can include sales databases, or Integrated Development Environments (IDE), and can compile resources hosted by Cloud computing platforms such as Amazon.com, Microsoft, Google, and Yahoo. The Cloud-based model [10] entails that the Web-based tools are operating systems, and are client machineagnostic. A further advantage of the Cloud model is the flexibility of implementation.

Grenze ID: 02.ICCTEST.2017.1.165

III. Cloud Based IOT Platform[1]

As we know IoT system consists of three main parts sensors, network connectivity and data storage applications. The same has been shown in figure-1. As shown in the figure, Sensors in the IoT devices either communicate with gateway devices or directly with the central server for data storage various Sensors for different applications are used in various IoT devices as per applications such as temperature, Power, Humidity, proximity etc Gateway takes care of various interfaces and one gateway can handle multiple technologies and multiple sensors. The typical wireless technologies used widely are wimax,WI-FI, Zigbee, Zwave and RFID, NFC and others.IOT supports both IPv4 and IPv6 protocols. With the support of IPv6 of t 128 bit long IP address length, there are enough addresses available to growing demand of IoT devices. The unique feature of IoT is DTN (Delay Tolerant Networks) which controls of large variable delay requirement of IoT based networks compare to traditional computer networks. IoT providers offer services with varied QoS , pricing, memory, CPU and battery consumption.

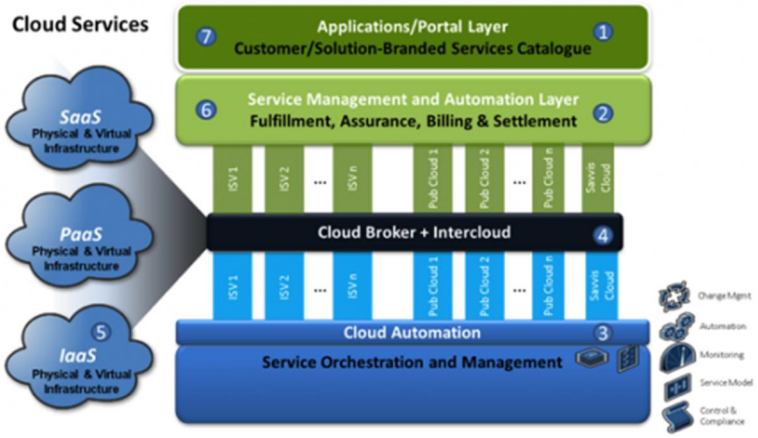

Fig1 :Cloud Based IOT platform

IV. Characterstics Of Iot

A. Enormous scale

The number of devices that need to be managed and that communicate with each other will be at least an order of magnitude larger than the devices connected to the current Internet

B. Connectivity

Connectivity enables network accessibility and compatibility

C. Sensing

D. Dynamic Changes

The state of devices change dynamically, e.g., sleeping and waking up.

E. Hetergonity

The devices in the IoT are heterogeneous as based on different hardware platforms and networks.

F. Safety

Securing the endpoints, the networks, and the data moving across all of it means creating a security paradigm that will scale and also securing our personal data.

V. Data Storgae Impact In Iot

The impact on storage is high in IOT. large amount of data is available to store. Machine generated data comes in two distinct types, creating two different challenges. First, there is large file data, such as images and videos captured from smart phones and other devices. This data type is typically accessed sequentially. The second data type is very small, for example, log-file data captured from sensors. These sensors, while

small in size, can create billions of files that must be accessed randomly. It used to be that a data center would have only one of these data types, They were either busy in capturing image-based data or not. however, data centers must deal with both data types, and the two different storage systems. one designed for large-file sequential I/O and other for small-file random I/O. Historically, image-based data has typically been placed on large-capacity NAS systems, but they are seeing a shift to object-based storage, at scale. Sensor data, usually stored on high-performance NAS systems, is moving to all-flash arrays, primarily to allow faster analytics.

A. Types Of Data In The Internet Of Things

We have categorized the data into the following areas：

1. Radio Frequency Identification address/unique identifiers.
2. Descriptive data, positional and environmental data sensor data
3. Historical data
4. Physics models-that are template for reality
5. state of actuators and command data for control

VI. DATABASE ISSUES IN THE INTERNET OF THINGS[5]

In this section we set out the areas of challenge for database management in the IoT and point out where developments in traditional databases and other areas may provide the solution

A. Size, Scale and Indexing

The size and scale of the data in the IoT will be vast. Data will need to be managed via responsible local ownership. Local owners will decide which data and services to make available to the global network. Thus, the IoT may operate on more than one level: private and public. Users may join groups for access to certain privately owned data or may, on the other hand, access data publicly available over the public Internet. There may be differences in quality of data depending on ownership and level of care. Gradually trust and reputation systems will provide information to users on the quality of the data.

B. Query Languages

Current popular query languages in database systems rely on structured data. Structured Query Language (SQL) is the most prominent example. Over the last few years, however, there have been proposals for query languages for semi-structured data, which is more typical of the data held on the Internet [22-28]. The quantities of data are so vast that it would be unrealistic to expect any sort of uniform structure, except perhaps that of the loosest variety, to be imposed on the IoT. xtensible Markup Language (XML) offers a means of representing less structured as well as structured data, together with some level of self description.

C. Process Modeling And Transactions

It is likely that most processes will be developed and supplied as services on the IoT. Service Oriented Architecture (SOA) is becoming an important means of supporting interoperability in web-based systems [9].The central idea is that independent out its offer services in a uniform manner, which other users can then take up. Thus implementation details are hidden from the users of the services. Application processes will typically be made up of a num-ber of lower level transactions. Transactions in turn will be made up of lower level operations or services. Therefore, the question of transaction processing in the IoT arises. In the traditional database systems the matter of concurrent transaction processing has been handled
through the maintenance of ACID properties through time stamping, locking, and a two-phase commit. ACID properties are atomicity, consistency, isolation, and durability. A transaction must complete in its entirety or not at all, a transaction must leave the database in a consistent
state, transactions should not show other transactions, and intermediate results and changes made by a transaction must be permanent. In distributed database systems a two-phase commit is used to preserve consistency. All participating sites must confirm their readiness to commit before the commit command is issued by the coordinating site and written to the database log. It has been recognized that the ACID properties do not web transaction processing well(32,33].This is because the individual web services are essentially autonomous and must independently preserve consistency.

D. Homogeneity And Integration

In the context of databases the areas of Heterogeneity and integration have been researched since the 1980s, once it was considered useful to achieve interoperability across heterogeneous systems [40-44].Considering that one might have a personnel system stored at one company in a relational database system, and in another company a similar system might be held in a network database system or even a different relational database system, questions arise as to how to integrate such data. Various solutions have been offered[45-49].Some promising solutions suggest the use of a canonical data model, for instance a functional or binary data model [50,51]. However, it seems that often the solutions offered do not warrant the efforts needed to achieve them

E. Time Series Aggregation

Time series aggregation is an interesting area, which has been noted as raising challenges in various application domains It has been recognized that inappropriate time aggregations can give rise to spurious causality.[7][8]the problem revolves around the ability to select the optimal sampling period for continuous data. Trade-offs include processing time and storage space against accuracy and realistic representation

VI. IOT DATABASE REQUIREMENTS[12]

There are many factors to keep in mind when choosing a database for an IoT application, and they do not always align with the needs of other more traditional enterprise databases. Some of the most important considerations are scalability, ability to ingest data at sufficient rates, schema flexibility, integration with analytics tools and costs. A database for IoT applications must be scalable. Ideally, IoT databases are linearly scalable so adding one more server to a 10 node cluster increases throughput by 10%. IoT databases will usually be distributed unless the application collects only a small amount of data that will not grow substantially. Distributed databases can run on commodity hardware and scale by adding new servers instead of swapping out a server for a larger one. Distributed databases are especially well suited for IaaS clouds since it is relatively easy to add and remove servers from the database cluster as needed.

An IoT database should also be fault tolerant and highly available. If a node in the database cluster is down, it should still be able to accept read and write requests. Distributed databases make copies, or replicas, of data and write them to multiple servers. If one of the servers storing a particular data set fails, then one of the other servers storing a replica of the data set can respond to the read query. Write requests can be handled in a couple of ways. If the server that would normally accept a write request is down, another node in the server can accept the write request and forward it to the target server when it is back online. IoT databases should be as flexible as required by the application. NoSQL databases -- especially key-value, document and column family databases -- easily accommodate different data types and structures without the need for predefined, fixed schemas. NoSQL databases are good options when an organization has multiple data types and those data types will likely change over time. In other cases, applications that collect a fixed set of data -- such as data on weather conditions -- may benefit from a relational model. In-memory SQL databases, such as MemSQL, offer this benefit.

A.Technical Considerations

NoSQL database serve as the primary data source for the intended online application

- Is NoSQL database operate as an analytic data source or easily interface with and support Hadoop operations
- Is NoSQL database handle or seamlessly integrate with enterprise search software?
- Can NoSQL database provide workload isolation between online, analytic, and search operations in a single application?
- NoSQL database is safe
- Data is safe?
- Does the NoSQL database provide a robust security feature set?

B. *Cloud Considerations For Nosql Databases*

- The amount of information that currently resides only in the cloud is small, but that's about to change The cloud promises many things: transparent elasticity and scale, higher availability, simplified data distribution, easier manageability and more. However, it should be noted that while many database

vendors claim their database is “cloud ready”, what that oftentimes means is that you can easily install and run an instance of their database on a cloud provider.

- The following are a suggested set of questions to ask any NoSQL database provider being considered for the cloud:
- Does the database provide transparent elasticity with expansion or contraction being possible without downtime?
- Can extra capacity be realized from scaling out in the cloud, and if so, how much benefit will be obtained?
- Can the database easily make use of a cloud provider’s multiple availability zones so that continuous availability can be achieved in the event of one or more zone’s failure?
- Does the database offer security features that protect data in the cloud?

Does the NoSQL vendor provide management tools for managing and monitoring the database on the cloud provider?

VII. DISCUSSION ON NOSQL DATABASE

This section address why NOSQL is used in IOT compare to sql and what are the NOSQL database categories and their performance in IOT

A. Why Nosql?

There are different kinds of data that are structured, unstructured and semi-structured and hence RDBMS systems are not designed to manage these types of data in an efficient way. NoSql databases comes in to the picture and are capable to manage it . People today are use different kinds of methodology and if you talk about the code velocity and implementation level, people wish to execute various activities at the same time. Talking about the traditional way, it was quite difficult in terms of taking care of the database and writing your application according to the database.

B. Challenges Of Nosql

Flexible data models: NoSQL databases have far more relaxed or even nonexistent data model restrictions. NoSQL Key Value stores and document databases allow the application to store virtually any structure it wants in a data element. Even the more rigidly defined BigTable-based NoSQL databases (Cassandra, HBase) typically allow new columns to be created without too much fuss.The result is that application changes and database schema changes do not have to be managed as one complicated change unit

Maturity: NoSQL alternatives are in pre-production versions with many key features yet to be implemented.

Support: NoSQL systems are open source projects, and although there are usually one or more firms offering support for each NoSQL database

Analytics and Business Intelligence **:** NoSQL databases offer few facilities for ad-hoc query and analysis. Even a simple query requires significant programming expertise, and commonly used BI tools do not provide connectivity to NoSQL.

*Administration***:** The design goals for NoSQL may be to provide a zero-admin solution, but the current reality falls well short of that goal. NoSQL today requires a lot of skill to install and a lot of effort to maintain.

*Expertise***:** large number of developers throughout the world are familiar with RDBMS concepts and programming. But every NoSQL developer is in a learning mode. So it is not easier to find experienced RDBMS programmers or administrators than a NoSQL expert.

C. Comparison Of Sql And Nosql

THE BELOW TABLE SHOWS THE MAIN DIFFERENCE BETWEEN SQL AND NOSQL DATABASES [11]

	SQL Databases	NOSQL Databases
Types	Single type ie SQL database	Different types including key-value stores, wide-column stores, document and graph databases
Examples	Microsoft SQL Server, MySQL ,Oracle Database, Postgres	HBase ,MongoDB, Cassandra, , Neo4j
Schemas	Data types and Structure remain fixed and entire database has to be altered to store a new data item	Dynamic in nature with data validation rules .new fields can be added by an application on run .dissimilar data can be stored as SQL row wise in a table
Scaling	vertical scaling	Horizontally scaling:
Supports Transactions	yes it support transactions updates to entire or not at all	at certain levels (e.g., document level vs. database level)
Data Manipulation	Select, Insert, and Update statements.	object-oriented APIs
Insert query Example	`INSERT INTO book ( `ISBN`, `title`, `author`)VALUES ( '978762461256', 'Stack', 'Biltz');`	`db.book.insert({ ISBN: "9786775256", title: "Stack ", author: "Biltz"});`
update query Example	`UPDATE book SET price = 19.99 WHERE ISBN = '9780992461256'`	`db.book.update( { ISBN: '9780992461256' }, { $set: { price: 19.99 } });`
Consistency	it can be configured for strong consistency	some offer Strong consistency and eventually depends on product

D. Nosql Database Categories[13]

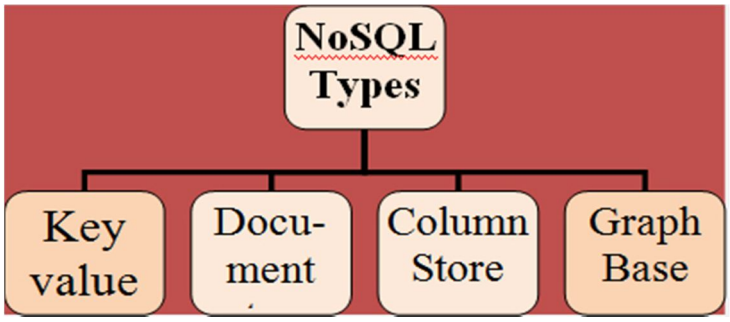

Fig 2 Categories of NOSQL Database

Key Value Store NoSQL Database
it has a Big Hash Table of keys & values
Document Store NoSQL Database
It stores documents made up of tagged ele ments
Column Store NoSQL Database
Each storage block contains data from only one column, (Ex- HBase, Cassandra)
Graph Base NoSQL Database
A network database that uses edges and nodes to represent and store data. (Example- Neo4J)
Below table shows the comparison of NOSQL database with their functional model and their data representation and language used for data retrieve and their performance.

TABLE II: COMPARISON OF NOSQL DATABASE CATEGORIES

	Types of NoSQL database			
	Key/value Based	*Column Based*	*Document Based*	*Graph Based*
Operational /functional models	Redis memCacheDB etc	Cassandra Hbase etc	MongDB Couch Base etc	Orient Neo4J etc
Data model	Key–Value Store	Column-Oriented Store	Document-Oriented Store	Graph Database
Query language	Any programming Language	Java API REST API	Java ,Xquery	C++,java,phyton etc
Popular use cases	Caching queuing Distributed Information Keeping live information	Keeping unstructured non volatile information scaling	Nested Information JavaScript friendly	Handling complex relations information Modeling and handling classification
Performance	High	High	High	Variable
Scalibility	High	High	Variable	Variable
Functionality	Variable	Minimal	Variable	Graph Theory

E. Choosing A Nosql Database

Nosql Databases is different from SQL database from its features and functions , but the below criterion helps narrow the field for deployments. They are

- The data model: This involves the type of data need to store and its starting or ending format.
- The data scale expectations: This involves how large an application is expected to grow and the data scale support that will be needed. Some NoSQL databases are main memory and do not scale out across multiple machines, whereas others like Cassandra scale linearly across many machines.
- The data distribution model: how widely data needs to be distributed, whether to support multiple geographic regions, for disaster recovery purposes, or not.

VIII. CONCLUSION

The NoSQL is a good fit for IoT, as we face the challenge of dealing with huge volumes of data over time. For businesses that wish to scale their IoT implementations and make use of the data that these networks create, NoSQL solutions are a better fit than RDBMS options. The Internet of Things remains a key topic for standardization and attracts interest from industry, public authorities and end users. GSC members reviewed current standardization activities focused on specific applications and use cases, such as smart cities and intelligent manufacturing. They also explored how IoT can help address global challenges such as electricity access in the developing parts of the world. GSC agreed on the importance of increasing outreach to both end users and industry stakeholders to accelerate the development and adoption of future proof IoT standards.

REFERENCES

[1] Challenges Keyur K Patel1and Sunil M Patel2 ,"Internet of Things-IOT: Definition, Characteristics, Architecture, Enabling Technologies, Application & Future",International Journal of Engineering Science and Computing, May 2016

[2] Jaime Lloret , Jesus Tomas and Alejandro Canovas, Lorena Parra, ",An Integrated IOT Architecture for Smart Metering", IEEE Communications Magazine ,pp.50-57,Dec 2016

[3] Liana Stanescu ,Marius Brezovan and Dumitru Dan Burdescu , "Automatic mapping of NOSQL database to MongoDB", Computer Science and Information Systems (FedCSIS), 2016 Federated Conference ", Sep 2016.

[3] Katarina Grolinger1 , Wilson A Higashino1and Abhinav Tiwari1 and Miriam AM Capretz, "Data management in cloud en-vironments: NoSQL and NewSQL data stores" ,Springer Journal of cloud computing, 2013

[4] Sakr S, Liu A, Batista DM, Alomari M," A survey of large scale data management approaches in cloud environments. IEEE Commun Surv Tutorials , pp. 311–336., 2013.

[5] Hoboken, New Jersey, USA: John Wiley & Sons Beyer MA, Laney D", The Importance of "Big Data",29 Sep 2013
[6] Sakr S, Liu A, Batista DM, Alomari M," A survey of large scale data management approaches in cloud environments. IEEE Commun Surv Tutorials , pp. 311–336., 2013.
[7] Abiteboul S, Manolescu I, Rigaux P, Rousset M-C, Senellart P ",Web Data Management.", Cambridge University Press; 2012.
[8] Bughin J, Chui M, Manyika J," Clouds, big data, and smart assets: Ten tech-enabled business trends to watch", pp. 1–14, 2010.
[9] Tudorica BG, Bucur C, " A comparison between several NoSQL databases with comments and notes", 10th International Conference IEEE ,pp. 1–5,2011.
[10] Hecht R, Jablonski S," NoSQL evaluation: A use case oriented survey.", Proc 2011 Int Conf Cloud Serv Computing , pp.336–341,2011.
[11] D. Ceballos, and M. Sorrosal, "Time Aggregation Problems in Financial Time Series", MS'2002 International Conference on Modelling and Simulation in Technical and Social Sciences, Working Article of Institute of Time Nature Exploration, pp. 243-52.,2010
[12] Stonebraker M, Madden S, Abadi DJ, Harizopoulos S and Hachem N and Helland P," The end of an architectural era: (it's time for a complete rewrite).", Proc 33rd Int Conf Large Data Bases ,pp. 1150–1160,2007.
[13] A.P. Levich (ed.), On the way to understanding the Time Phenomenon: the Constructions of Time in Natural Science,Part I, World Scientific, New York, 1995.
[14] Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I: Cloud computing and emerging IT platforms: Vision, hype, and real-ity for delivering computing as the 5th utility. Future Gen Computer Syst ,pp.25(6):599616,2006.
[15] E. Newcomer, and G. Lomow, "Understanding SOA with Web Services, Addison Wesley", 2005.
[16] Chang F, Dean J, Ghemawat S, Hsieh W, Wallach D, Burrows M, Chandra T, Fikes A, Gruber R: Bigtable: ,"A distributed structured data storage system." 7th OSDI 2006, 26,pp. 305–314,2006.
[17] https://www.mongodb.com/nosql-explained
[18] http://www.techrepublic.com/blog/10-things/10-things-you-should-know-about-nosql-databases/
[19] http://internetofthingsagenda.techtarget.com/feature/Find-the-IoT-database-that-best-fits-your-enterprises-needs
[20] https://www.digitalocean.com/community/tutorials/a-comparison-of-nosql-database-management-systems-and-models
[21] http://www.cloudamize.com/insights/blog/cloud-comparison-cheat-sheet-aws-vs-google-vs-microsoft-azure/