IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Nosql: An Overview And Its Comparative Study With Relational Database About To Generate The Most Frequent N-Gram

Arti Sharma, research scholar,

Dr. Rajesh Soni, Associate professor,

Department of computer science and application,

Bhupal Noble's University, Udaipur, RAjasthan

Abstract

Larger data volume of various applications development leads to the today's IT growth, now a day it is observed, with Relational database we cannot work on large data to maintain high volume data base due to the strict constraints on data structure, data relations and so on. Various formats of different industries including unstructured data has to be stored and processed in the databases. Hence, NoSQL types are the solutions for various issues of large data base. Largely NOSQL (Not Only SQL) is soughtafter due to the advantages such as horizontal scalability and flexibility. This paper, discusses about different types of NoSQL database, including MongoDB, Couchdb, HBase, Cassandra, etc.

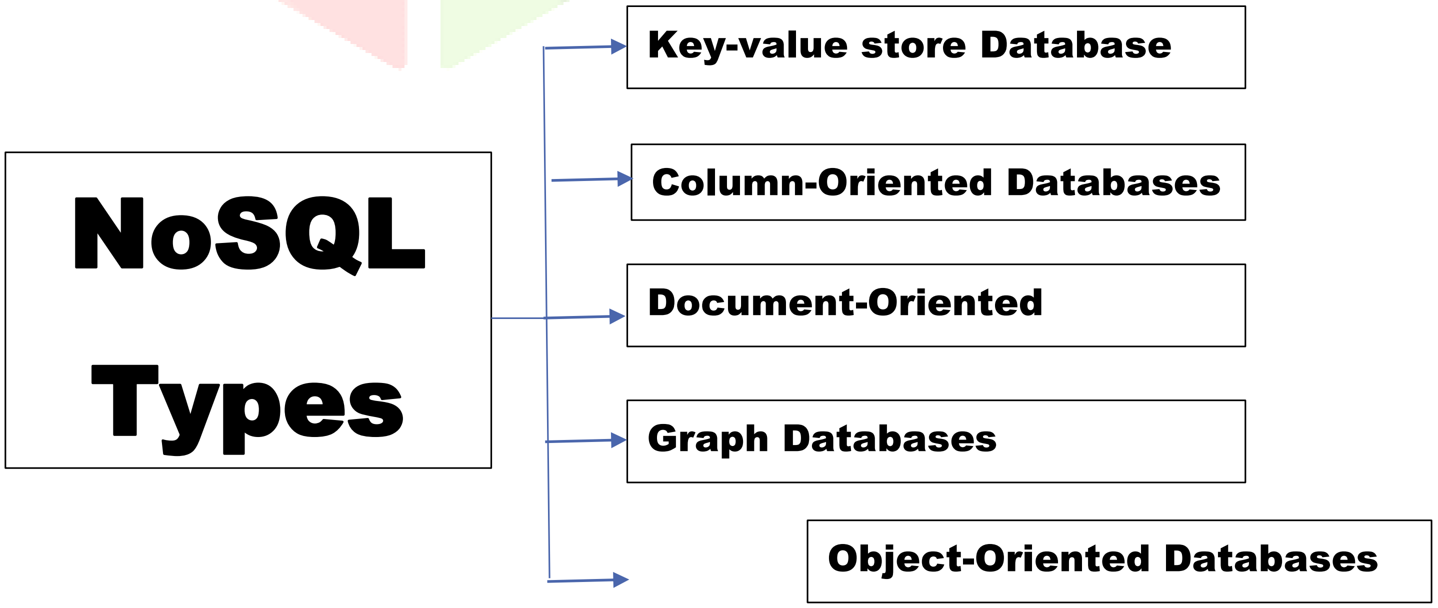
Keywords - Relational Database, Big Data, NoSQL, MongoDB, Couchdb, HBase, Cassandra.

Introduction

NoSQL is a free and open-source, scattered, extensive column stores database management system intended to handle large amounts of data across many product servers, providing high obtainability and accessibility with no single point of failure. It is the easiest truly big-data database that can scale and replicate data globally in a master-less configuration. A NoSQL database delivers a mechanism for storage and recovery of data that is demonstrated in means other than the tabular relations used in relational databases. NoSQL databases usually understood by engineers as 'not only SQL databases' neither 'no SQL', it is an alternative to the most widely used relational databases. As the given name proposed, it is a substitute for SQL that uses in such a way that the SQL is co-existed.

The technology of relational databases did not meet the storage needs of this new demand. In order to deal with this demand, the Big Data technology emerged. Big Data as a technology that has three characteristics: Volume, Variety and Velocity, also known as the 3 Vs: Volume: Storing and analyzing a huge volume of data; Variety: Handling different types of data, from structured to semi-structured ones or non-structured; Velocity: Handling with the demand by high speed that data are accessed and stored. The NoSQL databases (Not only SQL) came to fill the 3Vs demands of the Big Data. NoSQL as a new generation of databases allowing a high performance and fast processing of information in large scales. With the consolidation of WEB 2.0, a sub-area of Data Mining became popular: Text Mining. It involves the extraction of previously unknown knowledge from texts. Among its main applications we may highlight: sentiment analysis and inferring the gender and age of a user from text of social networks. As a subarea of Data Mining one of the main challenges of Text Mining is the building of its input variables. Among the main variables used by his area the highlight are the N-grams, which consider the number of times than n terms appear together in a specific order in a text. Although NoSQL is frequently used to storage WEB 2.0 data, the developers of Text Mining solutions have no access to NoSQL databases when they build their solutions. For instance, the PAN@CLEF competition, one of the main ones in this area, releases data for building solutions using XML (Extensible Markup Language) files containing the posts of users. A question that arises in this scene is: "which is the most effective way of obtaining N-grams from XML files?". The objective of this paper was to answer this research question. In order to do that, an experimental study was executed comparing the two main technologies of data storage.

NoSQL and Relational databases. The study used the database oftheinternationalcompetitionPAN@CLEF2013.Themetric used for performance evaluation was the time in seconds that each approach needs to obtain the most frequent N-grams. Results obtained show that the performance of NoSQL databases is superior that the one of the relational databases.



1. Key-value store database: The key-values store database is very well organized and devastating model. It can easily communicate to API (Application Programming Interface). API is an application programming interface is a set of procedure definitions communication protocols and tools for building software. The key-value data can be stored in an appearance that schema may not be needed and data can be stored in tables based on their data types of the programming language or an object. The data has as its main feature and divided into two parts, a string which signifies the key and the actual data which is to be associated as value thus generating a key-value pair.

The values are stored in hash tables where the keys are the indexes which makes it faster than RDBMS. Hence the data model is very unexacting and easily manageable. The data is stored with NoSQL Types high extensibility over reliability and so the querying features like joins and collective processes have been excluded. One of the drawbacks for key-value store database is since there is no schema, it is difficult to create custom views. Key-values data storage mainly used in user's session or shopping cart, to get the list of favorite products saved, websites, forums, online shopping, etc. Some of the examples of a key-value store database are Amazon Dynamo DB, Riak, etc.

Amazon DynamoDB: It is a NoSQL database offered by Amazon which is highly predictable and scalable.

• Fast and flexible NoSQL database service for all applications that need consistent, single-digit millisecond latency at any scale.

• Supports both document and key-value store models.

• Flexible data model, reliable performance, and automatic scaling of throughput capacity make it a great fit for mobile, web, gaming, ad tech, IoT, and many other applications.

• Fully Managed. No longer required to have any concern about database management tasks.

• Fine-grained Access Control. It integrates with the AWS Identity and Access Management for fine-grained access control.

• Event Driven Programming. Integrates with AWS Lambda to provide Triggers which enables to architect applications that automatically react to data changes.

• Highly Scalable. Automatically scales capacity up or down, an application request volume increases or decrease.

• DynamoDB Accelerator (DAX) is a fully managed, highly available, in-memory cache. Can reduce response times from milliseconds to microseconds, even at millions of requests per second.

Riak: Riak is a scattered NoSQL key-value data store that advances high accessibility,liability tolerance, effective effortlessness, flexibility, and adjustability. In addition to the open-source version, it originates in a reinforced enterprise version and a cloud storage version.

• Unmatched flexibility beyond typical 'high availability' contributions.

• Advanced technology to ensure data accuracy and never lose data.

• The immense scale on product hardware.

• Common code basis with true multi-model support.

• Allocates data across the cluster to ensure fast performance and fault-tolerance.

• Multi-cluster replication ensuring low-latency and robust business steadiness.

• Faster reads and writes making it easier to store, query, and analyze time and

location data.

It can be used for the following purposes:

• Handling individual information of the user for social networking websites or

MMORPGs (Massively Multiplayer Online Role-Playing Games).

• To collect checkout or POS (Point of sales) data.

• Managing Factory control and Information systems

• Building Mobile Applications on the cloud etc.

Riak should be evaded for highly centralized data storage projects with secure, fixed data structures. Riak is used by Mozilla, AOL, and Comcast.

2. Column-Oriented Databases: Column-oriented database stores data in a table by column instead of rows. Column stores in NoSQL are composite row-column storage,unlike pure relational column databases. Although column-oriented database concept of column-by-column storage of columnar databases and columnar extensions to row-based databases, in NoSQL column-oriented stores, do not store data in tables. It stores data in enormously scattered designs and architectures. In a column-oriented database, each key is associated with one or more attributes (columns).

The column-oriented database provides high flexibility and scalability in data storage. These types of database are suitable for data mining and analytical applications. Some of the distinguished DBaaS providers using column-oriented Databases are mentioned below.

Cassandra: Apache Cassandra is the easiest truly big-data database that can scale and replicate data globally in a master-less configuration. What used to be in the hands of only the biggest in Silicon Valley is now available as a mature database to the masses. Originally created at Facebook after they studied Amazon's DynamoDB and Google's BigTable whitepapers, the Cassandra we know today is very different and has far surpassed its ancestors in feature set and has now become a popular wire-protocol for other databases such as ScyllaDB, YugaByte, and Azure's CosmosDB.

Apache Cassandra is written in Java language. Why it is chosen to be written in Java may be because the security is a prime concern it is developed in Java rather than in C++. Another key reason could be Performance. It might be slower at the startup, but once the code is ready and in running state it is way faster as compared to C++. Java code is continuously optimized by the JVM and in that consideration, it appears faster to C++. It may have other reasons as well such as advanced memory optimization or efficient garbage collection.

There are different flavors of Apache Cassandra available in the market,

• ScyllaDB is an open-source distributed NoSQL datastore which was intended and

designed with Apache Cassandra while achieving expressively higher throughput and lower latencies. It’s written in C++.

• YugaByte DB is a transactional and high-performance distributed database for building largescale scattered cloud services. It also supports APIs which are Cassandra compatible and Redis compatible, with PostgreSQL in the Beta stage. YugaByte DB core is written in C++, but the repository contains a Java-based code that is needed to run sample applications.

• DataStax Enterprise offers Apache Cassandra flavor in a database platform which is built knowingly for providing performance and availability demands of IOT, Web and Mobile applications. It gives organizations a safe always-on database that effects operationally simple when scaled in a single or across multiple data centers and in the clouds. Cassandra and DataStax Enterprise have helped the customers supporting multi datacenter and hybrid cloud deployments since the beginning. It is written in Java.

Let’s understand the merits of using Apache Cassandra :-

Cassandra is a unique platform to handle huge amount of unstructured data at scale. If you’re trying to make your relational database faster and reliable, Cassandra may be your ultimate companion. It combines the Amazon’s Dynamo storage system along with Google’s Bigtable model, offering the near-constant availability required to support real-time querying for web and mobile apps.

• Cassandra can handle even the most massive datasets.

• It can work as amazing, record-setting reliability at scale.

• Eventual consistency yields high availability.

• It offers Wide-column flexibility.

• It also offers minimal administrative tasks at scale.

• It offers easy setup and maintenance (does not matter how big the dataset that you are setting)

• Flexible parsing and wide column requirements.

• Not with multiple secondary indexes.

• It allows applications to write into any node anywhere and anytime.

• Automatic workload management and data balancing across the nodes

• Linearly scalable by just adding more nodes to the cluster.

On the other hand, it may not be a proven benefit if,

• Your app requires transactional operations,

• It requires dealing with financial data,

• Need dynamic queries against column data,

• Low latency requirement,

• Read exceeds write by a large margin

• On-the-fly aggregations & joins and so on...

It is hard to find which large-scale organization does not use Cassandra nowadays. When dealing with distributed databases, it is always the key requirement to identify how the data and the workload will be distributed. Correspondingly, the data model must be correctly designed. For example,

• Not letting the partition key too large,

• A specific size of the tables,

• Keeping an ideal same partition size, etc.

The most important point to highlight is even though distributed databases falls under the category of the database, however, treating this application to behave like a traditional relational database may incur excessive performance degradation and it may break the application as well.

So, we must have to be careful while designing the application.

Bigtable: Goggle's big table is high performance and compressed data storage system which is built on Google file system, Chubby Lock Service, SSTable and few other Google technologies. Bigtable also inspires Google Cloud Datastore, which is accessible as a part of the Google Cloud Platform. Bigtable is used by many Google applications such as web indexing, Google Maps, Google Book search Google Earth, Google Code, YouTube, Gmail, etc. Google has developed its own database to increase scalability and performance. Google's Spanner RDBMS is covered on an application of Bigtable with a Paxos group for two-phase commits to each table. Google F1 was built using Spanner to substitute an operation based on MySQL. Bigtable is one of the perfect examples of a wide column store. Bigtable is designed to add thousands of machines and it is easy to add more machines.

• Three main mechanisms: the library, the master server, and many tablet servers. The library is connected to many clients, the master server manages the schema changes, tablet servers manages the tables.

• It is not a relational database and can be well defined as a light, scattered multi-dimensional organized map.

• Bigtable is used to scale into petabyte mode.

• When table size portends to raise beyond a definite limit, the tablets may be compressed using the algorithm BMDiff and the Zippy compression algorithm.

• It provides steadiness, reliability, fault tolerance and tenacity.

3. Document-Oriented Databases: Document-oriented databases also called document store databases is a subset of a type of NoSQL database. It is designed for storing, retrieving and handling document-oriented information. Document-oriented databases are characteristically a subclass of the key-value store, another NoSQL database concept.

This offers great performance and horizontal scalability choices. The data stores in a form of documents and the storage are like records but the data are more flexible as there are no uses of schemas. Documents can be stored in the form of PDF. JSON, XML etc. Document storage in the document-oriented database is complex because it stored in key value pairs also known as key-document pairs. Documents may have special characters' in it. It's easy to fetch the data if documents are portioned across some documents. Some of the document-oriented databases provide are MongoDB, CouchDB, etc. MongoDB: an open source distributed document database, highly optimized for JSON. Apache CouchDB: an open source, Erlang based, database with a RESTful HTTP API.

MongoDB: MongoDB is a document-oriented database program that can be implemented on multiple computer platforms. This is classified as a NoSQL database program. 10gen software company started developing MongoDB in 2007 as a planned platform as a service product and during 2009 it was initially released as an open source development model. During 2013 the 10Gen changed the name to MongoDB. C++ was used to develop MongoDB as a high performance and effective database.

• MongoDB is highly optimized for JSON, it stores data in flexible JSON documents that means the columns may vary document to document and the data structure may be reformed over time.

• Easy to work with as the object mapping is done by the document model in the application code.

• The real-time aggregation, the indexing, and queries give significant ways to access and examine the data.

• MongoDB is a scattered database at its fundamental, so high obtainability, horizontal scaling, and topographical circulation are built in and can be used easily.

• Absolutely free to use, the versions are released before October 16, 2018, are available under the AGPL. All versions are released after October 16, 2018, including patch fixes for prior versions, are published under the Server-Side Public.

• MongoDB provides end-to-end security.

• Management tools are available for automation, monitoring, and backup.

• Fully elastic database as a service with built-in best practices.

• High accessibility through built-in replication and failover.

CouchDB: Apache CouchDB is open-source database software that emphases on ease of use and having a scalable architecture. This also has document-oriented NoSQL database architecture. CouchDB was first introduced in 2005. Later it became the Apache foundation project in 2008. Each CouchDB is a group of independent documents. Unlike a relational database, CouchDB never stores data and relationships in tables. Each document preserves its own data and uses its own schema. Like MongoDB, CouchDB is developed using C++. The data storing can be done in JSON format and database with a RESTful HTTP API.

• http-based REST interface using which documents can be easily created and managed.

• Easy to set up with multiple nodes. Easy repetition of a database across multiple server instances.

• Gets data in the form of JSON format and to store the data it takes only a few minutes.

• The data stores in a format that space is not wasted leaving empty fields in the documents.

• When the frontend editing option is available, it is possible to set up an application very fast for loading and managing data.

• CouchDB has flexible schema designs, fast indexing, and retrieval of data.

4. Graph Databases: Graph databases are NoSQL databases which use to store data as a graph. This data model encompassed of vertices, which is an object such as a person, place, pertinent section of data and edges, which signify the connection between two nodes. The graph also entails of possessions related to nodes. The associations permit data in the store to be connected straightly and mostly recovered with one operation. Graph databases hold the relations between data as a priority. Querying relations within a graph database is fast because they are eternally stored within the database itself. Relations can be instinctively imagined using graph databases, making it beneficial for deeply inter-connected data. In the graph database, every node consists of a direct pointer which points to the adjacent node. Millions of records can be traveled using this method.

Graph databases offer schema-less and effective storage of semi-organized data. The queries are articulated as traversals, therefore creating graph databases are quicker than relational databases. While the graph model clearly lays out the addictions between nodes of data, the relational model and other NoSQL database models connect the data by implicit ways.

Some of the Graph databases and the language it supports are listed below –

• AllegroGraph (Language: C#, C, Common Lisp, Java, Python)

• Amazon Neptune

• AnzoGraph (Language: C#, C)

• ArangoDB (Language: C++, JavaScript, .NET, Java, Python, Node.js, PHP, Scala, Go, Ruby, Elixir)

• DataStax Enterprise Graph (Language: Java)

• InfiniteGraph (Language: Java)

• JanusGraph (Language: Java)

• MarkLogic (Language: Java)

• Microsoft SQL Server 2017 (Language: SQL/T-SQL, R, Python)

• Neo4j (Language: Java, .NET, JavaScript, Python, Ruby)

• OpenLink Virtuoso (Language: C, C++)

• Oracle Spatial and Graph; part of Oracle Database (Language: Java, PL/SQL)

• OrientDB (Language: Java)

• SAP HANA (Language: C, C++, Java, JavaScript & SQL-like language)

• Sparksee (Language: C++)

• Sqrrl Enterprise (Language: Java)

• Teradata Aster (Language: Java, SQL, Python, C++, R)

Amazon Neptune: Fast, Reliable, Fully-managed graph database service that makes it easy to build and run applications that work with highly connected datasets.

• Fast, reliable, fully-managed graph database service makes it easy to build and run

applications with highly connected datasets.

• The core of Amazon Neptune is a purpose-built. High-performance graph database

engine that is optimized for storing billions of relationships and also to query the graph with milliseconds latency.

• Highly available, with read replicas, point-in-time recovery, continuous backup to

Amazon S3, and replication across Availability Zones.

• Secure, with support for encryption at rest and in transit.

• Supports open graph APIs for both Gremlin and SPARQL and provides high performance for both graph models and their query languages.

• Highly available, durable, and ACID (Atomicity, Consistency, Isolation, Durability) compliant.

• Fully Managed. User needn't worry about database management tasks such as hardware provisioning, software patching, setup, configuration, or backups.

5. Object-Oriented Databases: In the object-oriented database, the data or information is stored in the form of objects. It is different from the Relational database as this is not table-oriented. Object-oriented database management systems are also known as object database management systems syndicate database competences with object-oriented programming language competences. Object-oriented database management systems let the object-oriented programmers build the product, load them as objects and modify the existing objects to make a new object. Accessing data in the object-oriented database are comparatively faster as an object can directly be retrieved from its pointer.

Some object oriented databases are intended to work fine with object-oriented programming languages such as C++, C#, Python, Ruby, Delphi, Perl, JavaScript, Java, Visual Basic .NET, Objective-C, and Smalltalk, etc. This database can be used in the applications relating to complex object relationships, modifying object structures when the application describes collections. Scientific research, telecommunication, software that analyze, optimize and design are some places where object-oriented databases are mainly used. The scalability becomes problematic when the physical memory size exceeded in Object-oriented database. Some of the Object-Oriented Databases are listed below –

• db4o

• GemStone/S

• InterSystems Caché

• JADE

• ObjectDatabase++

• ObjectDB

• Objectivity/DB

• ObjectStore

• ODABA

• Perst

• OpenLink Virtuoso

• Versant Object Database

• ZODB

Advantages of Object-Oriented Databases**:**

• The object-oriented database lets the data, information, components, products to distribute effortlessly.

• This allows integration easily the databases, operating systems, spreadsheets, other systems like AI, objects, and applications, etc.

RELATED WORKS

Several works performed comparisons between relational and NoSQL databases. However, according to the literature research performed in this study, no one compares the performance of those two databases for obtaining the most frequent N-grams. Following will be described works that did some type of comparison between the two database approaches.

The authors did a comparative study of the operations of inserting, excluding and querying between NoSQL and relational databases. The MongoDB, RavenDB, CouchDB, Casandra, Hypertable and Couch base databases were used as representatives of the NoSQL approach. Microsoft SQL Server Express was used as representative of the relational approach. The study found that not all NoSQL databases had a better performance regarding the same operations implemented in relational database. Besides, the study found that there is a considerable difference of performance between NoSQL databases. The authors compare the NoSQL database based in documents MongoDB with the relational database PostgreSQL regarding execution time of select and insert operations. The study indicates superiority of the MongoDB in the insertion text. The PostgreSQL time was 1106 seconds while in MongoDB it was 17.78seconds for inserting 100000 records. In the text of complex search, the PostgreSQL took 278438 seconds while MongoDB needed only 51.71 seconds to perform the same task. The authors compared the NoSQL database based in documents MongoDB with the relational database MySQL in an application of information management. Results showed that MongoDB is faster than MySQL in insert and query expressions. However, the authors inform that the lack of references about MongoDB slowed down the process of developing the solution. The authors compared the NoSQL database based in documents MongoDB with the relational database Oracle. The authors conclude that if the objective is speed and scalability, NoSQL is the most indicated one. On the other hand, if speed is not the priority and the objective is to better structure the data, the relational approach is indicated. The authors compared the NoSQL HBase with the relational database MySQl. Times of reading and writing large amounts of data were evaluated. The study pointed that the reading and writing times were smaller using HBase. The authors highlight scalability, availability, performance and fault tolerance as advantages of HBase regarding MySQL. The authors compared the NoSQL and relational approaches in the distributed management of RDF (Resource Description Framework) data. HBase and MySQL Cluster were used as representatives of the NoSQL and relational approaches respectively. The study showed HBase as superior for querying large sets of RDF data.

ABOUT THE MAIN PROBLEM

The process of building variables is one of the oldest and still challenging problems of the Data Mining area. emphasizes that variables in Text Mining may be represented as a vector of features that captures potentially relevant characteristics from the text. Among the most common ones extracted from text it is possible to highlight the N-grams. They represent the occurrence of N items in sequence in a given text. The items can be letters or words. For example,

for the sentence "Attitude is a little thing that makes a big difference", if N=3 (words), then the N-grams would be: 1) "Attitude is a" 2) "is a little" 3) "a little thing" 4) "little thing that" 5) "thing that makes" 6) "that makes a" 7) "makes a big" 8) "a big difference" The quantity of N-grams built in a database, generally, is very large, because of that it becomes unfeasible to use all N-grams for applying machine learning algorithms, because they can create a phenomenon known in literature as the "curse of dimensionality". In order to overcome this problem, it is common to select the N-grams that will be used as variables. This selection should use some criterion, such as: the frequency of N-grams. In this way, first all N-grams of each text are obtained and then the X most frequent N-grams are selected. A practical problem that developers of Data Mining solutions meet is how to obtain the most frequent N-grams in an effective way regarding time. Generally, data available to developers are in files representing a sample from all texts. In order to help developers in this task, this work performed a comparative study between the two most frequent approaches for obtaining N-grams.

Nowadays every organization deals with a massive amount of records from a variety of sources at revolutionary speeds. Relational databases are sometimes ineffective for businesses processing and investigating the vast amount of multifaceted and unstructured data. As NoSQL is schema-less or fixed schema model databases, it is very efficient to handle a large amount of data and it is set to real-time data accessibility data model. Mostly all organizations are swamped with loads of data every second from a variety of sources retrieving from the internet. With this data, validation can be done for making the best or most effective use it and for making future predictions. Using NoSQL real-time analytics are much faster, response times can be under a minute for all complex queries. In SQL databases tables are tied with the primary, foreign keys whereas in NoSQL different data model can be used to deal with the gigantic amount of data. When a user wants to use key-value pairs, the key-value databases can be used, for data pointers graph databases can be used, more nodes can be added to the cluster which is easily scalable instead of using big machines.

COMPARED APPROACHES

The two main technologies of data storage were selected, aiming to help developers of Text Mining solutions to decide the most effective way of obtaining the most frequent N-grams. The methodologies for obtaining the most frequent N-grams using NoSQL and relational databases will be presented next.

A. NoSQL database: - The NoSQL database selected for this study was HBase. It is the no relational database based in the BigTable technology of Google. HBase is a component from the Apache Hadoop framework that uses the Hadoop files system, the HDFS (Hadoop Distributed File System), for data storage. Hadoop allows the distribution and processing of large data amounts in computer clusters using a simple programming model and offering high fault tolerance. Hadoop offers tools enabling an easier extraction of relevant information from a database. One of those tools is Hive. Hive is a warehouse software facilitating the reading, writing and handling of large data sets stored in a distributed way using SQL commands. It allows doing a query of the most frequent N-grams in a table using a simples command, similar to a SQL command. For obtaining N-grams using HBase. Initially it was necessary to install the tools used by Hadoop: Java, ssh (secure shell) and rsync (a tool for synchronizing files in computer systems). Then, it was necessary to install and configure Hadoop. Next, to install and configure HBase for using the HDFS of the installed Hadoop. Derby was needed, because it is used to save the metadata stored by Hive. After all Hadoop tools were installed and configured, a Java program for extracting the posts that were in several XML files was used for inserting the data in the HBase database. A table was created in Hive and connected with the table where the posts where inserted in HBase. Finally, a query was executed in Hive in order to get the most frequent N-grams.

B. Relational database: - The relational database selected in this study was MySQL. Figure 3 illustrated the step followed in order to get the most frequent N-grams using a relational database in this study. First, the following tools were installed: Java, Python and the MySQL database. Java was installed for using the program that creates a file with all posts. This program was adapted from the program used in the NoSQL approach for inserting data in HBase. Python was installed for allowing the use of the NLTK (Natural Language Toolkit) library, in order to obtain the N-grams. NLTK is a leading platform for building Python programs that work with human language data. After installing the tools, the sequence of activities is: 1) executing the Java program for creating a file with all posts; 2) running a program written in Python which obtains the N-grams and inserts them in a new file; 3) inserting the file with all N-grams in a MySQL table and; 4) running a SQL command in order to get the most frequent N-grams.

Conclusion

NoSQL can be used by many advanced applications. NoSQL makes machine learning must faster. Thousands of transactions every second can easily be monitored using NoSQL so it’s useful when working with fraud detections of banking transactions. Also, tentatively 2.5 quintillion bytes of data that generate from social media, climate data, innovative pictures and footages, the conversation of data, and that's just the starting. For these scenarios, numerous kinds of elements e.g. Pictures, Video, and Audio are integrated and stored in the database. Various NoSQL databases are scalable to handle the information. With the growth of Big Data, the operation of NoSQL invention is growing faster among all web organizations and creativities. Assistance includes elastic design, scaling and well switch over convenience.

NoSQL databases are serving well web organizations to influence their analytic goals in the fast-paced world. Activists of NoSQL preparations are fast that it delivers faultless handling and boosted implementation comparative to traditional relational databases. NoSQL address the problems of overwhelmingly focused organizations viewing to regulate to varying client requirements and varied growing markets. This technology beats outlooks at processing major unstructured data and the organization joins most popular open source products like Cassandra, MongoDB, Redis, etc. NoSQL also provides a less-expensive substitute for data load and retrieval. If we consider the pros and cons for both NoSQL and SQL, the best approach will be to combine both for additional impulsion research horizons and make it more productive in the future.

This work presented a comparison between relational and NoSQL database approaches for obtaining the most frequent N-grams. The comparison was performed using a database from an important international competition, considered as a benchmark in the area. As experimental methodology, tStudent's paired one-tailed test was applied in the performance measured by the time needed for each approach for obtaining the N-grams. The study showed that the NoSQL approach overcomes, in a statistically significant way, the relational database with a confidence level of 95%. The main contribution of this work is to indicate for the developers of Text Mining solutions the use of NoSQL technology for obtaining the most frequent N-grams. The main limitation of this work was to use only one representative for each technology. Thus, as a future work the authors pretend to expand this work using other representatives of the NoSQL and Relational database approaches.

References

[1] Fig:1 Retrieve from https://www.kdnuggets.com/2016/07/seven-steps-understanding-nosql databases.html

[2] Cassandra Vs RDBMS, retrieved from https://www.javatpoint.com/rdbms-vs-cassandra

[3] Retrieve from https://www.mongodb.com/what-is-mongodb

[4] Mukherjee, S. (2019). Popular SQL Server Database Encryption Choices. arXiv preprint arXiv:1901.03179.

[5] Mukherjee, S. (2019). Benefits of AWS in Modern Cloud. arXiv preprint arXiv:1903.03219.

[6] Mukherjee, S. (2019). How IT allows E-Participation in Policy-Making Process. arXiv preprint arXiv:1903.00831.

[7] Mukherjee, S. (2019). How Stakeholder Engagement Affects IT Projects. International Journal of Innovative Research in Science, Engineering and Technology, 8(3).

[8] Fatima, Haleemunnisa & Wasnik, Kumud. (2016). Comparison of SQL, NoSQL and NewSQL databases for internet of things. 1-6. 10.1109/IBSS.2016.7940198.

[9] Chakraborty, Moonmoon &amp; Excellence, Operations. (2019). Supply Chain &amp; Inventory Management. 10.6084/m9.figshare.7824107.

[10] Mukherjee, Sourav. (2019). Overview of the Importance of Corporate Security in business.10.15680/IJIRSET.2019.0804002.

[11] Mukherjee, Sourav. (2019). How stakeholder engagement affects IT projects. 10.15680/IJIRSET.2019.0803265.

[12] Chakraborty, M. (2019). Fog Computing Vs. Cloud Computing. arXiv preprint arXiv:1904.04026.

[13] Mukherjee, Sourav. (2019). SQL Server Development Best Practices. International Journal of Innovative Research in Computer and Communication Engineering. 10.15680/IJIRSET.2019.0803266.

[14] Mukherjee, S. (2019). Indexes in Microsoft SQL Server. arXiv preprint arXiv:1903.08334.

[15] Yoon, Byoung-Ha; Kim, Seon-Kyu; Kim, Seon-Young (March 2017). "Use of Graph Database for the Integration of Heterogeneous Biological Data". Genomics & Informatics. 15 (1): 19–27. doi:10.5808/GI.2017.15.1.19. ISSN 1598866X. PMC 5389944. PMID 28416946

[16] Author Craig Kerstiens (April 4, 2019), retrieve from Postgres and superuser access https://www.citusdata.com/blog/

[17] Chakraborty, Moonmoon. (2019). Planning, Control Systems and Lean Operations in Information Technology. 10.6084/m9.figshare.7886138.

[18] Chakraborty, Moonmoon. (2019). Managing Risk, Recovery & Project Management. 10.6084/m9.figshare.7886141.

[19] Chakraborty, Moonmoon. (2019). OPERATION IMPROVEMENTS & QUALITY MANAGEMENT IN HEALTHCARE Operation Improvements & Quality Management in Healthcare. 10.6084/m9.figshare.7886144.

[20] Adhikari, Mainak & Kar, Sukhendu. (2014). NoSQL databases. 109-152. 10.4018/978-14666-6559-0.ch006.

[21] Deka, G.C.. (2014). NoSQL databases. 10.4018/978-1-4666-5864-6.ch008.

[22] R. Neto, P. J. Adeodato, and A. C. Salgado, “A framework for data transformation in credit behavioral scoring applications based on model driven development,” Expert Systems with Applications, vol. 72, no. 1, pp. 293–305, 2017.

[23] T. Jauhiainen, H. Jauhiainen, K. Lind´en et al., “Discriminating similar languages with token-based backoff,” in Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects, 2015.

[24] J. Fan, Y. Fan, and Y. Wu, High-Dimensional Classification. World Scientific, 2011, ch. 1, pp. 3–37.

[25] L. Havrlant and V. Kreinovich, “A simple probabilistic explanation of term frequency-inverse document frequency (tf-idf) heuristic (and variations motivated by this explanation),” International Journal of General Systems, vol. 46, no. 1, pp. 27–36, 2017.

[26] Y. Li and S. Manoharan, “A performance comparison of sql and nosql databases,” in Communications, Computers and Signal Processing (PACRIM), 2013 IEEE Pacific Rim Conference on. IEEE, 2013, pp. 15–19.

[27] C. Politowski and V. Maran, “Comparac¸ ao de performance entre postgresql e mongodb,” in X Escola Regional de Banco de Dados. SBC, 2014, pp. 1–10.

[28] Z. Wei-ping, L. Ming-Xin, and C. Huan, “Using mongodb to implement textbook management system instead of mysql,” in Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on. IEEE, 2011, pp. 303–305.

[29] A. Boicea, F. Radulescu, and L. I. Agapin, “Mongodb vs oracle-database comparison.” in EIDWT, 2012, pp. 330–335.

[30] H. Ding, Y. Jin, Y. Cui, and T. Yang, “Distributed storage of network measurement data on hbase,” in Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on, vol. 2. IEEE, 2012, pp. 716–720.

[31] C. Franke, S. Morin, A. Chebotko, J. Abraham, and P. Brazier, “Efficient processing of semantic web queries in hbase and mysql cluster,” It Professional, vol. 15, no. 3, pp. 36–43, 2013.

[32] D. Carstoiu, E. Lepadatu, and M. Gaspar, “Hbase-non sql database, performances evaluation,” in in Computer Science (1986), Master in Computer Science (1990), and PhD in Computer Science. Citeseer, 2010.

[33] A. Hadoop, “Hadoop,” 2009.

[34] E. Loper and S. Bird, “Nltk: The natural language toolkit,” in Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1, ser. ETMTNLP ’02. Association for Computational Linguistics, 2002, pp. 63–70.