***Advanced Databases***

***Research Report***

***INDIVIDUAL ASSIGNMENT***

Name: - **Muhammad Abu Bakar Sani**

Student No: - **x17112044**

Course: - **BSHC 3**

**No SQL Databases**

**Motivation: -**

NoSQL databases are commonly known as Non SQL or Non Relational databases. It provides an apparatus for retrieval and storage of information or data. It is commonly used in big data and real time web apps. Prior to the last few years there was an explosion of several NoSQL solutions and products. Several largest web firms are using solutions based on Non SQL i.e. Twitter, Facebook, Google, Amazon etc. If you have bulk amount of data to handle, RDBMS is not a good choice because performance is not up to the required level and not extremely scalable in spread or distributed environment. Its important to remember that Non SQL Databases are complement in database field but not a replacement for RDBMS. That’s why it also called Not Only SQL. NoSQL database follows high availability e.g. Facebook, Google. We can keep various types of data filed in a same group(collection) because there is no such structure well-defined to be followed that’s why transactions are eventually consistent e.g. MongoDB. NoSQL doesn’t lock the whole or entire transaction when transactions interface with each other e.g. Amazon book store. NoSQL use replica. We create similar copies of data on various nodes, within the system if machines crash, the rest of the computers remain normal (unaffected) and work continues. It is highly scalable.

**Here are the three main types of NoSQL Databases**.

**key-value store databases: -** A data is represented as a collection of key-value pairs. It is also known as associative array organized into rows. key-value store databases store data as a hash table with a unique key and pointer to specific item of data. It allows data storage and retrieval through keys, similar to traditional hash table. Whenever the data needs to be retrieved really fast and would be queried by exact parameters, key-value stores are used. A logical group of keys is bucket. But it doesn’t physically group the data. There can be identical keys in different buckets. Because of the cache management (accompany the mapping) performance is enhanced to a great degree. To read a value you ned to know both key and bucket because the real key is hash. Example, Key represent the country name, while Value list the address.

|  |  |
| --- | --- |
| Key | Value |
| “Ireland” | “B-25, Palmerstown, D – 20” |

key-value store databases, handle size and are good at dealing out with constant stream with low latency marking them perfect for: -

High scale session management.

Profile and user performance stores.

Product recommendations.

Ad servicing.

Rarely updated data but can efficiently work as a cache.

Here are some of the popular key-value databases: -

Riak, Redis, Couchbase Server, Berkeley DB

**Column-oriented databases: -** Rather then as rows of data, data is stored in cells grouped in columns of data. Columns are grouped into column families. Read and write is done using columns rather than rows. Column families contain a virtually unlimited number of columns that can be created at run time or the definition of the schema.

Most RDBMS stores data in rows, but columns provide fast search, access and data combination. Relational Database stores different rows in different places on disk (single row as continues disk entry), on the other hand Columnar databases stock all cells to a column as a continuous disk entry to make access and search faster.

For Example: - To query the “Titles” from a bunch of a million books will be a thorough task while using relational databases as it will go over to each location to get item titles. On the other side, with just one disk access, title of all the books can be attained.

Here are key benefits of Column-oriented databases: -

* + High performance on combinational queries (like COUNT, SUM, AVG etc).
  + Extremely efficient data compression
  + Rapid data loading & true scalability for Big Data.
  + Available to many 3rd party BI analytic tools.
  + Fairly simple administration systems.

The top-known examples are Google’s BigTable and HBase. BigTable, has high performance, compressed and proprietary data storage system maintained by Google

**Document store databases: -** It is also known as document oriented, aggregate or document database. It practices or uses document oriented model to store information. It stores each record and its linked data within a solo document. Each document holds semi structured data that can be used against various query, problems and as analytical tool of the DBMS. Data, which is a collection of key value pairs is compressed as document store quite (Key value store). It provides some encoding and structure of the managed data. XML, JSON (Java Script Object Notation), BSON (binary encoding of JSON objects) are some common encodings standard. The following example show the data values: -

|  |
| --- |
| {officeName:” Gill Ventures”,  {Road: “B-23, City: “Punjab”, State:” UP”, Pincode:”2013”}} |

Document oriented database are well suited for a extensive variety of use case. Document store databases are using in following sectors: -

**Web Application: -** **ECommerce** web applications, Web analytics, Blogging platforms etc

**User’s Granted Contact: -** Chat Sessions, Tweets, Blog post, Comments

**Computing / Networking: -** Sensor data from mobile devices, Log Files, Realtime analytics

Here are some of the examples of the leading document store DBMS’s i.e. MongoDB, DocumentDB, MarkLogic etc

**Here is an example of each three types of NonSQL Databases: -**

**Riak: -** Is one of the most popular example of a key-value data store databases that provides full availability, fault / Problem tolerance, scalability and functioning simplicity with reduce / map, HTTP’s, JSON (Java Script Object Notation) and Rest queries making it perfect for web apps. Written in Erlang. It mostly comes in two versions supported enterprise version and cloud storage version. Its first version was released on 12 August 2009. It has fault / Problem tolerant data repetition and auto data delivery across the clusters for resilience and performance. It implements the ideologies from Amazon (S3) Dynamo paper with highly encouragement from the Brewer’s Theorem. Riak’s MapReduce has an additional goal i.e. increase data locality. Its often much more efficient to process the large dataset. In training your MapReduced job code is likely less than 10kB’s, so it is more effective to send the code to the GB’s of data being processed than to stream gigabytes (GB’s) of data to your 10kB’s of code. It has authorized drivers for Ruby, Java, Erlang and Python.

**Here are some of its key features:** -

Storage opportunities: - Values / Keys, or both can be stored in memory disk.

Tunable constancy: - Option to select between strong consistency and eventual for each bucket.

Predictable latency: - With hashing, riak distribution data across nodes can provide potential profile. Even in case of multiple node failure.

Queries: - It provides Rest-full API’s through HTTP and protocol buffers for PUT, GET, POST and DELETE roles. More complex queries are also likely to include secondary indexes. MapReduce has natural support for both java and Erlang.

**When should you use Riak: -**

* + You need extremely fast and reliable NoSQL databases.
  + When you need private cloud storage for non-huge (small) files.
  + When you need to play with your database via simple Rest interface.
  + When you need JavaScript based mapreduce engine.

**HBASE: -** It is a column-oriented NoSQL DBMS that runs on top of Hadoop Distribution File System (HDFS). It provides a fault tolerant way of storing data sets, commonly in many large data use cases. HBase is most suitable for the processing of Real time data, random read or write access to large volumes of data. Structures query language (SQL) and applications written in Apache Avro, Rest and Thrift are not supported by HBase. It isn’t a relational database store at all. Like a typical Apache MapReduced Application HBase apps are also coded in Java. This system is designed to scale linearly. It is made of a set, of standard tables with Attributes and tuples like old-style database. Each table must/should have a primary key and all requests attempt to HBase tables must use this primary key. Avro support a rich set of original data types like integers, binary and strings data. And also, number of complex data types including arrays, maps and records. Sort order can also be defined for data.

For high performance coordination HBase relies on ZooKepper. It is assembled into HBase. But if user running a production bunch, it is suggested that you have a devoted ZooKepper cluster that integrated with your HBase cluster. HBase works very well with Hive to enable fault tolerant big data applications.

HBase was designed to scale across cluster. It is not a traditional RDBMS.

**Here are some of its key features:** -

Failover: - HBase tables are replicated for failover.

Wide-columns: - Information (Data) in HBase is stored in a table like format through the ability to store millions of rows with millions of columns. To allows spreading of row values on different cluster nodes, columns can be grouped into column families.

Unstructured or semi-structured data: - Like RDBMS, Data stored in HBase doesn’t require to be fit into a rigid schema. That makes it ideal for semi-structured or Unstructured data.

**Apache CouchDB: -** It is a document oriented database and within each document fields key value maps are stored. Fields can either be a simple key/value pair, list or map. It is implemented in Erlang, JSON is used to store data, HTTP for an API and Query language is JavaScript using MapReduce. Unlike relational DB a CouchDB also doesn’t store relationships and data in tables. It was first released in 2005. Each data that is kept in the database is giving document level unique identifier (id) as well as a (rev) figure for each change that is made and saved back to the database. It has a cross-platform compatibility. Because of Multiversion concurrency control (MVCC) CouchDB doesn’t lock the database file during editing. Conflicts are left to the application for resolution. To resolution them, it generally involves first merging data into one of the documents, then deleting the stale one. CouchDB also includes Document level ACID semantics with eventual constancy, replication and MapReduce. One of its unique features is multi-master replication, which permits to scale across machines to build high performance systems. Fauxton (A Built-In web application) helps with administration.

**Here are some of its key features:** -

HTTP API: - Because of HTTP, all items have a unique URI that’s get exposed. It uses following HTTP’s methods POST, PUT, GET, DELETE for the basic operations.

Eventual Consistency: - CouchDB guarantees to be able to provide potential tolerance and availability.

Fast indexing and retrieval.

For document insertion, updates, retrieval and deletion REST-like interface is used.

Document format in JSON.

Multi libraries for user’s choice of language.

**Advantages and Disadvantages of NoSQL Databases.**

**Advantages: -**

* + Scalability: It has elastic scalability (Horizontally scalable). RDBMS is not easy to scale out. To take the advantage of new nodes, NoSQL’s is making for transparent expansion. NoSQL Databases are intended to work with lower cost hardware’s. In real world it is a better fit where upward scalability is swapped with outward scalability.
  + Big data application: For an organization, it’s is a need to store enormous amount of data, transaction rates are growing from recognition. RDBMS have grown too much to meet the increasing needs, but realistically its very tough to use single RDBMS to control such amount big amount of data. NoSQL databases can easily handle this amount of data.
  + Economy: NoSQL databases are much more economical then Relation databases because RDBMS requires luxurious (expensive) storage systems and servers to work efficiently. On the other side, NonSQL databases can easily be connected to cheap hardware clusters.

**Disadvantages: -**

* + Administration: On ground reality is much different because the end goal for NoSQL databases design was to offer a product that would need no administration. For both installation and maintenance, NoSQL databases still need a lot of technical skills.
  + Business intelligence and analytics: With the demand of modern-day web application (web 2.0), NoSQL databases were created and most features are meeting these demands. NoSQL databases provide few features for query ad-hoc and examination (analysis), where the demand of a data app extend beyond to the read, insert, delete and update cycle of a web. Most common BI tools that several companies depend on don’t offer connectivity to NoSQL databases. This problem is resolved by using some tools i.e. PIG or HIVE, to provide ad-hoc functionality for NoSQL databases.
  + Less Mature: First RBDMS was released in the market about 25 years ago. It means RDBMS have been in a market from quite a long time. With the passage of time RDBMS have matured to became luxuriously functional and stable system. On the other hand, most of the NoSQL databases have just made out of their pre-production stages. And there are many important features and functions that haven’t implemented yet.

**Advantages and Disadvantages of Relational** **Databases.**

**Advantages: -**

* + Data Structure: It is one of the biggest advantage of RDBMS because table format can easily be understandable by the user and make it simpler and easier to implement it for requests. Data organisation and data access are settled according natural structure.
  + Language: RDBMS supports a language i.e. SQL (Sequential Query Language) because its syntax is simple. The keywords and phrases it implements is in simple English language. This feature helps the user to learn and intercept. Sometimes RDBMS is recognized by to add NoSQL features and functions to this SQL awesomeness. There are variety of features that have been implanted into the RDBMS along with databases design. It additionally helps in performance improvement and development.
  + Maintenance: Maintenance is easier with RDBMS because it supports the database admins to repair, control, maintain, test and backup the database that place within their main system. Built in automatic tools within RDBMS helps in automating these functions.
  + Multi User Access: In RDBMS user access or entree is multiple, means more than a user can access a database at a time. User can utilize the integral transactions management and locking functionality to access the data, as the data is being changed or updated. This function mainly helps to prevent any crashes between users working on the same data.

**Disadvantages: -**

* + Cost: For relation databases the cost of setting it up and upholding a database system is relatively high. For setting up relation databases a special software is required and could cost a money. Non-programmers, might find difficulty to it up and get the programme running because they need to implement a number of products or goods to set up the database. For larger enterprises it would be imperative to get external help from an experienced programmer.

* + Abundance of information: Abundance in complexity or difficulty cause a downside to Relational databases. RDBMS’s are made for forming data by characteristics. Object RDBMS is a new type of database that categorise complex numbers, designs and multimedia products. There systems are designed to handle more composite applications and have capacity to scale.
  + Structured Limits: RDBMS impose limits on field length (Vertically scalable). Because increase the length of a column on a solo server requires increasing things like CPU, RAM or SSD. User have to specify the amount of data that he wants to fit into a field. Some inputs are briefer than the actual, this can lead to data loss.

As discussed above, NoSQL DBMS is different from the Relational DBMS (SQL). RDBMS have a table-based structure and is strict with its predefined scheme. On the other hand, NoSQL DBMS can be document based, key value pairs etc. It doesn’t require any predefined scheme that allow user to work more easily with unstructured data. Relational databases are vertically saleable, cost of CPU, RAM or SSD increases with the increase in the amount of data. The horizontal scaling nature of NoSQL databases makes it more cost effective.

According to both of their characteristics, NoSQL databases proves to be a better option for modern applications that have constantly changing, more complex data sets, requiring a flexible model that does not need to defined immediately. Most of the developers and organisations that prefer NoSQL databases are using Agile methodology because it allows them to go to the market and make changes faster. Unlike Relational databases, NoSQL can store and process data in real-time. NoSQL databases have many functions and features that Relational databases doesn’t have like Relational databases are not capable to control without tremendous cost, sacrifices of speed, agility etc. After all that comparison, it is proved that NoSQL database have much more dominant properties and specifications in both business and technical fields than Relation databases and in future most of the companies will start using it.

**Bibliography**

CoreView. (2019). NoSql Databases – Key-Value Store - CoreView. [online] Available at: https://coreviewsystems.com/nosql-databases-key-value-store/ [Accessed 13 Oct. 2019].

3Pillar Global. (2019). Exploring the Different Types of NoSQL Databases Part ii. [online] Available at: https://www.3pillarglobal.com/insights/exploring-the-different-types-of-nosql-databases [Accessed 14 Oct. 2019].

Annovate.blogspot.com. (2019). Motivation to NoSQL Database. [online] Available at: http://annovate.blogspot.com/2014/06/motivation-to-nosql-database.html [Accessed 12 Oct. 2019].

MongoDB. (2019). NoSQL vs SQL (Relational Databases). [online] Available at: https://www.mongodb.com/scale/nosql-vs-relational-databases [Accessed 17 Oct. 2019].

En.wikipedia.org. (2019). Riak. [online] Available at: https://en.wikipedia.org/wiki/Riak#Main\_features [Accessed 15 Oct. 2019].

Eck, R. (2019). An Overview of Riak: an Open Source NoSQL Database - Monitis Blog. [online] Monitis Blog. Available at: https://www.monitis.com/blog/an-overview-of-riak-an-open-source-nosql-database/ [Accessed 15 Oct. 2019].

Ibm.com. (2019). What is HBase?. [online] Available at: https://www.ibm.com/analytics/hadoop/hbase#targetText=HBase%20is%20a%20column%2Doriented,Distributed%20File%20System%20(HDFS).&targetText=An%20HBase%20system%20is%20designed,much%20like%20a%20traditional%20database. [Accessed 16 Oct. 2019].

En.wikipedia.org. (2019). Apache CouchDB. [online] Available at: https://en.wikipedia.org/wiki/Apache\_CouchDB [Accessed 17 Oct. 2019].

InfoQ. (2019). What is CouchDB and Why Should I Care?. [online] Available at: https://www.infoq.com/articles/warner-couchdb/ [Accessed 18 Oct. 2019].

Hadoop360. (2019). Advantages and Disadvantages of NoSQL databases – what you should know. [online] Available at: https://www.hadoop360.datasciencecentral.com/blog/advantages-and-disadvantages-of-nosql-databases-what-you-should-k [Accessed 18 Oct. 2019].

blogs, V. (2019). Advantages & Disadvantages of RDBMS - Best 10 Features of RDBMS | Acodez. [online] Web Solutions Blog. Available at: https://acodez.in/advantages-disadvantages-rdbms/#Disadvantages [Accessed 19 Oct. 2019].

Anon, (2019). [online] Available at: https://www.techwalla.com/articles/disadvantages-of-a-relational-database [Accessed 19 Oct. 2019].