International Journal of Computer Applications (0975 – 8887)
Volume 67– No.2, April 2013

NoSQL, a Solution for Distributed Database Management System

Renu Kanwar
M-Tech, Department of C.S
Govt. Engg. College, Ajmer
Ajmer (Rajasthan), India

Prakriti Trivedi
Asst.Prof, Department of C.S
Govt. Engg. College, Ajmer
Ajmer (Rajasthan), India

Kuldeep Singh
Senior Software Developer
Royal Bank of Scotland
Gurgaon (Haryana), India

ABSTRACT

The recent advance in distributed data management techniques are growing, today the world need databases to be able to store and process big data effectively, demand for very high-performance when reading and writing, these effects especially in large scale and high concurrency applications, such as search engines hence the traditional database limits itself for such complex requirements, therefore various types of non-relational databases that are commonly referred to as NoSQL databases which is abbreviation of " Not only structured query language". Their primary advantage which is worth emphasizing is that, unlike relational databases, they handle unstructured data. NoSql is the solution for use cases where ACID is not the major concern and uses BASE instead which works up on eventual consistency. Here introducing Coherence which is a technology developed by oracle which works on NoSQL Principles.

Keywords

NoSQL, Data-bottleneck, Coherence.

1. INTRODUCTION

While considering the current scenarios and the recent advances in cloud computing there are various outlooks where the traditional relational database limits itself considering the following aspects:

Big Data Storage: Large application like search engines require enormous amount of data to be stored and respond well with millions of data traffic.

Speed and Scalability: As the number of concurrent request increases there should be provision for easy expansion and up gradation.

Highly available and fault tolerant: The system should be available any ways and there shouldn't be single point of failure, it should provide fast data backup and recovery.

In other words there is requirement of applications which are highly scalable and can be manipulated according to ones needs and requirements. Therefore as day by day many applications have raised where the traditional database limits itself and there is a need for some reincarnation NoSQL databases have proved to be a solution, Internet is yet another example for the big data storage as it is the biggest database of the world hence for these unstructured data[6], a solution which bless the users by a wish by which they could design a database of their own according to their requirements, NoSQL are called so because they don't work on the traditional SQL protocols[8], Introducing Coherence which is a technology developed by oracle which works on NoSQL principles and make use of data partitioning in a very refined way that makes data scaling a child's play, it not only help in data partitioning but is fault tolerant and avoids single point of failure.

2. DATA BOTTLENECK

Traditional RDBMS encounters a very predominant problem of data bottlenecking which arises when there is large number of concurrent requests for a single application and system fails to respond to all the requests simultaneously. Considering the problem of data bottleneck there are some statistics of the traditional RDBMS that is to be rectified like scaling up the system through database clustering where data is made available at multiple sites thus it could be a solution and bottlenecking could be avoided but only to a certain extend as the replication of data is limited and could again face the same problem with the growing data numbers plus there is an important issue of data synchronization because a change at a single data site would require all the data sites to be updated and hence results into another problem of data distributed locking. To avoid the problem of distributed locking another appropriate solution is to shard the data repository into multiple data storage without replicating whole data like dividing the customer and product database so that both the problem of data bottlenecking and distributed locking is solved but a problem of data synchronization is encountered with such data sharding.

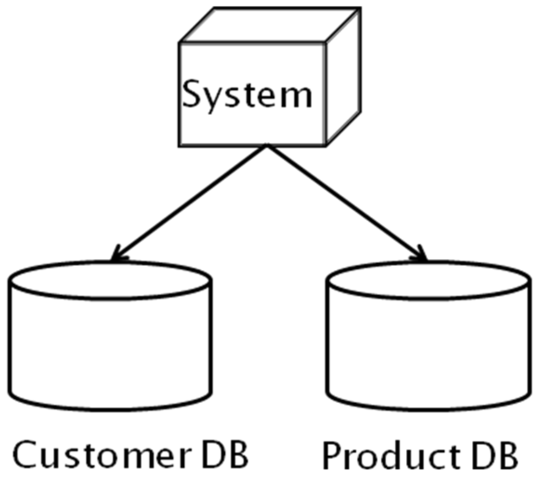

Fig 1. Database Sharding

Traditional RDBMS encounters a database schema which consists of a too many joins and relationships which makes the data storage a complex issue. Set of instructions which are not required for the application are often performed and hence increases the overheads on the system, The system have to deal with several tables and apply the possible joins in them which result in data locking and latching also following the strict rules of ACID which stands for Atomicity, Consistency, Isolation and Durability further adds up to the complexity that makes scaling of the database a complex task as when database need to be consistent various instructions are taken care of like buffer manager, hand code optimization etc whereas the overall useful work done is considerably low that increases unneeded complexities and overhead.

3. ACID->BASE

To avoid the un-needed complexities caused by the strict semantics of ACID NoSQL works on BASE [1].

The acronym BASE stands for

- Basically available
- Soft state
- Eventual Consistency

Hence the focus is on the eventual consistency that means while working the system is not obliged to maintain consistency at each state rather the belief is on making the system consistent at the end of any process means the readers will see writes as time goes on and then in a steady state the system will eventually return the last value which was written Clients therefore may face an inconsistent state of data as updates are in progress.

Table 1. ACID-BASE

ACID	BASE
Strict consistency	Weak constancy
Isolated	Availability first
Focus on “commit”	Best effort
Nested Transactions	Approximate answers
Conservative	Simpler
Difficult evolution (e.g. schema)	Fast, Easier evolution

Table 1 contrast between the two different semantics of ACID and BASE where the former laid emphasis on strict consistency and later on eventual consistency [12].

4. NOSQL ARCHITECTURE

NoSql architecture basically tries to achieve the shared nothing architecture, the term shared nothing means that when the data repository is being sharded the clusters should not be dependent on each other for any kind of updates any change inside any cluster should not affect the others.

To achieve this kind of architecture an extra scalable state is introduced which lies in the application layer and data is stored inside the memory RAM which is much more quick as compared to the disk, or it can be said that this is the caching layer which can be scaled out easily and also as data is stored In-Memory time is not wasted in the disk access.

• Ideally, this new layer should have the following characteristics: It should manage data as objects, because objects are what the application needs.

• It should keep these objects in memory, in order to improve performance and avoid the disk I/O bottlenecks that plague databases.

• It should be able to transparently load missing data from the persistent store behind it.

• It should be able to propagate data modifications to the persistent store, possibly asynchronously.

• It should be as simple to scale out as the stateless application layer.

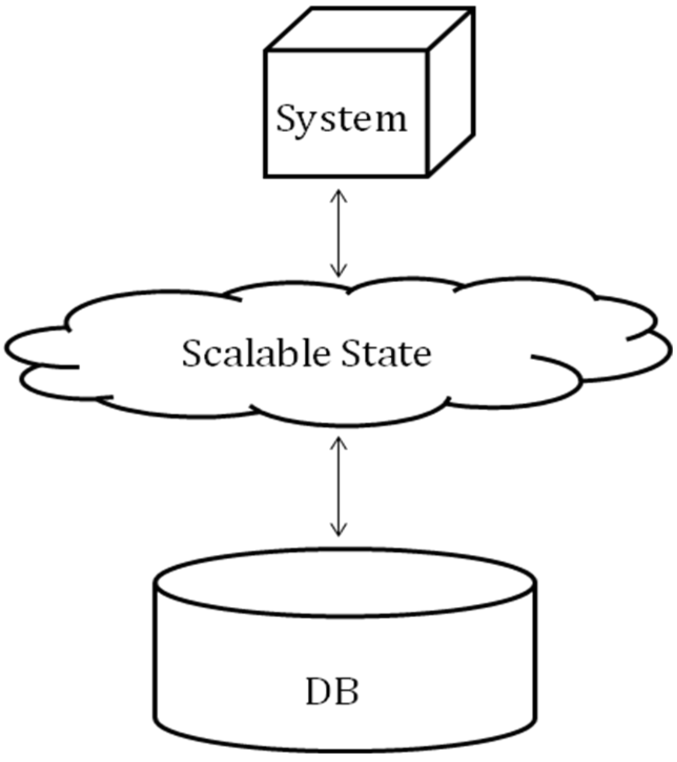

Fig 2.Shared nothing architecture

But the major problem with the In-memory data storage is that it is fragile and data could be lost. Therefore it is important to propagate the data modifications to the persistent store, possibly asynchronously, data is stored in the persistent data store asynchronously because the access to the disk takes time so the read write request of the user are entertained by the data stored in memory so that that the request are responded at high speed and then the data is stored inside the disk so that it won’t be lost if any miss happening occurs.

5. INTRODUCING COHERENCE

Coherence is the product of Oracle used in various banking multinationals, it can be thought as a cache, as it is designed for but it subsequently offers much functionality as query functionality, indexing etc. Coherence works on the same NoSql architectures were the clusters work as a cache which is called the Nodes and behind there is a persistent data store attached is the data is stored possible asynchronously.

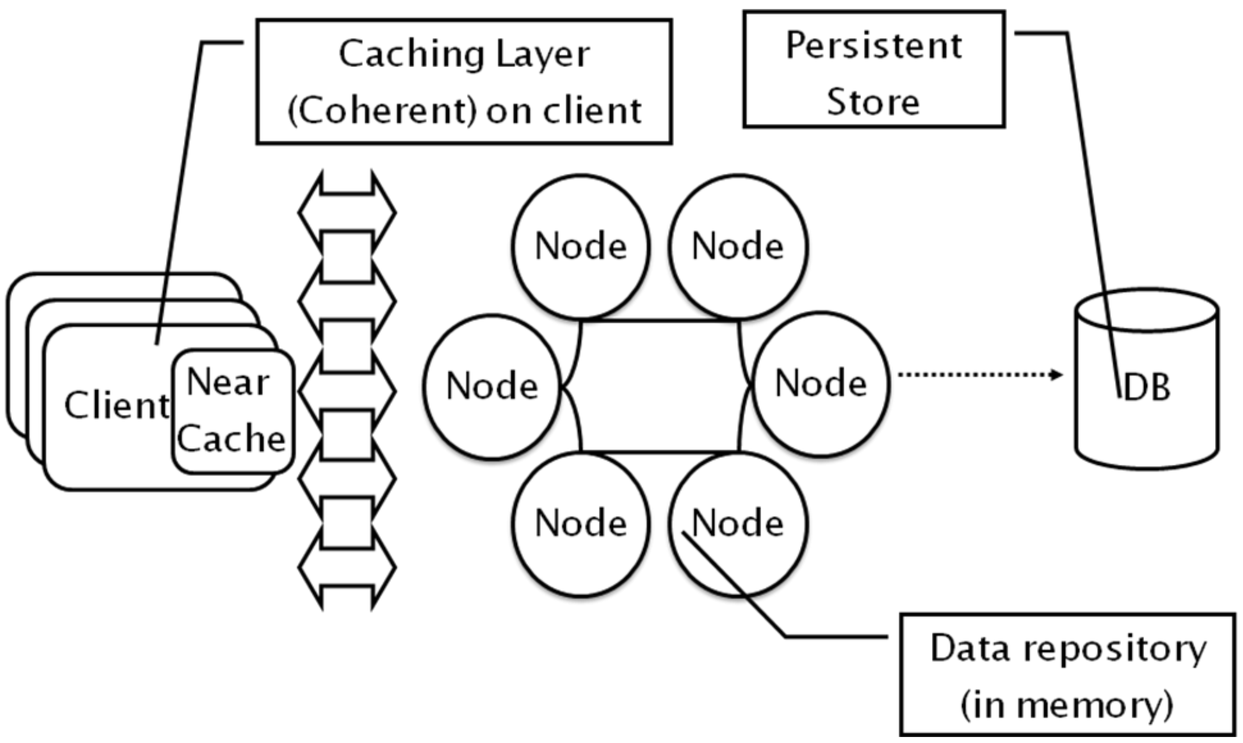
Caching Layer (Coherent) on client
Persistent Store
Node
Node
Node
Node
Node
Node
Client
Near Cache
DB
Data repository (in memory)

Fig 3.Coherence Architecture

There are three distinct layers in its architecture [10]; client, cluster, persistence data store, the cluster is sandwiched between the two layers of client and the persistent data source. The persistence data source is basically used for the data writes, it does not contributes to the data retrieval for that purpose the cluster in the centre is used which is usually pre-populated by the persistence data source.

Feature of Coherence

The three main features of Coherence is

- Fast
- Fault tolerant
- Scalable

Coherence is fast as it stores all the data solely in memory and therefore the disk access is not required which basically takes time and hampers the speed of the system[11]. Secondly the objects are always held in serialized form which allows coherence to skip the serialization step that cut down the extra load. Writes to the database are usually performed asynchronously, asynchronous persistence data store is important so that Coherence does not have to wait for disk access on a potential bottleneck source of data, It crafts the queries to run in parallel so that it can engages the entire hardware cluster simultaneously.

Coherence is Scalable because in RDBMS scaling is done through database clustering but with database clustering an obvious problem of distributed data locking is prevalent but with coherence these distributed locking problems are handled quite efficiently as all data is stored as key value pairs in small data clusters known as data nodes these nodes are quite flexible and can be increased and decreased easily [11],

According to the requirement the nodes are scaled the performance of coherence is comparatively better then RDBMS scaling.

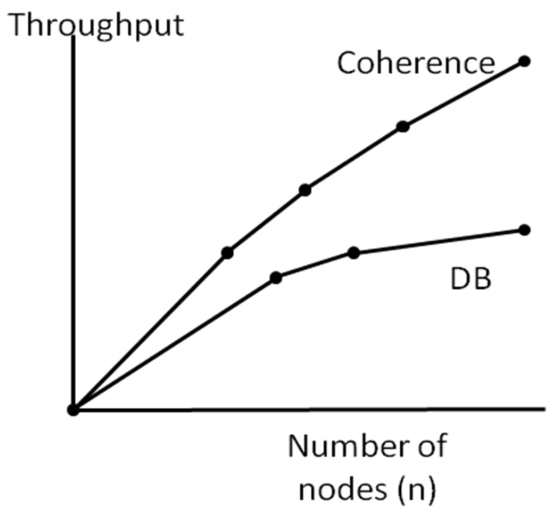

Fig 4.Scalability performance

Coherence is fault tolerant as well as highly available because with this product loss of single machine would not have significant impact on the operation of cluster [11]. Its resilience lies in the back up structure as loss of single node will result in failover to a backup copy held some were else in the cluster and all the operations which were going on will resume elsewhere in the cluster. It is the most powerful feature of the product plus it also detects node loss and effectively deal with it by adding of new nodes whenever the system require. Coherence says the more machines, the faster the failover will be.

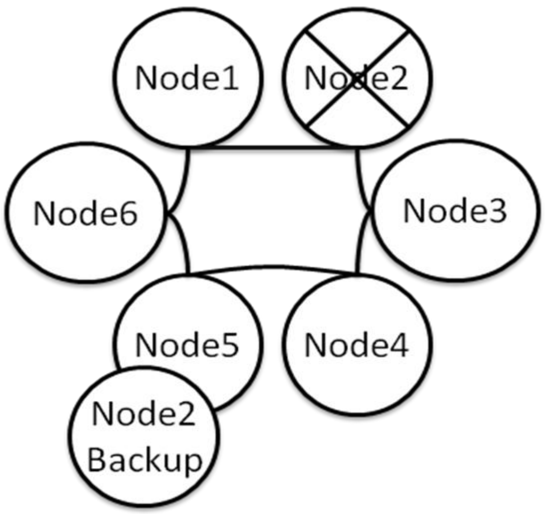

Fig 5.Failover and Backup

Coherence detects the Node which has died and redistributes the data among the remaining nodes. Cluster of nodes holding primary data locally the Back-up of primary data is distributed across all other nodes and hence a logical view can be obtained of all data from any node. All nodes verify health of each other by sending alert messages, In the event if a node is unhealthy, other nodes diagnose state of the unhealthy node as if for a long time if no messages are heard from a single node and a number of such alert messages are encountered the cluster term it as unhealthy. Unhealthy node is isolated from the cluster, and Remaining nodes redistribute primary and back-up responsibilities to all other healthy nodes. The distribution is completely fair and transparent.

6. CONCLUSIONS

Traditional database architectures have proved to be inappropriate for many use cases because in current scenario Speed and scalability are need of an hour. Therefore nowadays applications are shifting towards In-Memory data storage which could boost the data access and the system could look forward for databases which could work according to the use cases and NoSql is the solution for use cases where ACID is not the major concern. In this regard a very fine product of oracle that is Coherence provides us with three basic enterprise functionalities i.e. speed, scalability and fault tolerance. It has no single points of failure, it automatically and transparently fails over and redistributes its clustered data management services when a server becomes inoperative or is disconnected from the network. It Transparently redistribute the cluster load, Coherence runs on the java platform and is an example of NoSQL databases , It does however support an object based query language which is not dissimilar to SQL It is designed for very fast data access via lookups based on simple attributes, but have limitations of its own that when complexity increases and data needs to be centralized then coherence limits itself as data partitioning becomes difficult It is not suited for complex data operations or long transactions.

The combination of relational databases and NoSQL will bring a big change in data storage and by the popularity gained by NoSQL databases it seems as if after ten years may be the traditional database would get eradicated and NoSQL would take up its position.

7. ACKNOWLEDGMENT

I extend my thanks and gratitude to Mrs. Prakriti Trivedi (Asst. Prof.) my guide whose guidance, teaching and certain suggestion provide me the timely valuable input which enhanced my knowledge and motivated me for my work. I also wish to extent thanks to Mr. Kuldeep Singh who serves Royal bank of Scotland as a senior software developer for his contribution to the research work.

8. REFERENCES

[1] Bogdan George Tudorica, Cristian Bucur "A comparison between several NoSQL databases with comments and notes" 2011 IEEE Conference on Commerce and Enterprise Computing. April 6-7,2011

[2] A. Lakshman and P. Malik. "Cassandra: a decentralized structured storage system." SIGOPS Oper. Syst. Rev., 44:35–40, April 2010.

[3] Kai fan "Survey on NoSql databases" 10th IEEE /ACIS international conference on computer and information technology. 5-6 May 2011

[4] Jing Han, Meina Song, Junde Song, "A Novel Solution of Distributed Memory NoSQL Database for Cloud Computing" 10th IEEE/ACIS International Conference on Computer and Information Science. June 2011.

[5] Neal leavitt "will Nosql databases live up to their promise" IEEE computer society

[6] Hailing Zhang, Yang Wang, Junhui Han "Middleware Design for Integrating Relational Database and NOSQL based on Data Dictionary", 2011 International Conference on Transportation, Mechanical, and Electrical Engineering (TMEE), December 16-18, Changchun, China.

[7] Bucur, Cristian; Tudorica, Bogdan George, "Solutions for working with large data volumes in web applications", The Proceedings of the IE 2011 "Education, Research & Business Technologies" International Conference, 5-7 May 2011

[8] Michael Stonebraker, Uğur Çetintemel "“One Size Fits All": An Idea Whose Time Has Come and Gone" IEEE/ACIS International conference 2012.

[9] Lior Okman, Nurit Gal-Oz, Yaron Gonen, Ehud Gudes, Jenny Abramov," Security Issues in NoSQL Databases", 2011 International Joint Conference of IEEE TrustCom-11/IEEE ICESS-11/FCST-11

[10] Understanding distributed and in-Memory architectures http:www.benstopford.com/2011/08/14/distributed-storage-phase-change-memory-and-the-rebirth-of-the-in-memory-databases

[11] Understanding distributed and in-Memory architectures http;//www.benstopford.com/a-thing-of-the-past

[12] Cook, John D., "ACID versus BASE for database transactions", www.johndcook.com/blog/2009/07/06/brewer-cap-theorem-base/