International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Comparative Analysis of HBase Data Storage Model and RDBMS for Location based Application

Chandrakant Nitnaware[1], Amreen Khan[2]

Department of Computer Technology, YCCE Nagpur (India)

Abstract*: Relational database management system plays important role in the area of storage and management but have limitation on scalability in terms of data as well as number of clients. RDBMS is not suitable for spatial database application and very hard to maintain the system when data becomes too large. Column oriented database such as Hbase overcome this problem by storing large amounts of terabytes of data and providing the system to be fault tolerant and easily available. This paper compares geo-spatial data stores in Hbase Data storage model and RDBMS in terms of real time analysis and high insert throughput on latest location.*

Keywords*:* Hbase, NoSQL database, RDBMS, geo-spatial data, Location based application

1. Introduction

Millions of users continuously update their location on real time system is quite common now a days. So this has generates large volumes of data and leads to a various location based services[15] which customizes users experience based on location. To trace geographic location, Location based services uses the personal handset such as smartphone or navigation devices. These services may either use network-based technologies or embedded satellite navigation receivers to identify the geographic location. These location based services [6] use in various types of location based applications such as personal navigation, geo advertisement, tracking devices, location based social networking etc. Relational database management system (RDBMS) must handle such growing amounts of data while answering real time queries based on location[8]. However, relational database systems are not capable of handle high insertion rates, real time query and terabytes of data [1]. Whereas NoSQL databases like Hbase allows the system to sustain high insertion rates, large data volumes while ensuring the system to be fault tolerant and high availability [1].

NoSQL is a non-relational database management system which is different from traditional relational database management system. NoSQL database is suitable for efficient mass storage data, concurrent read write operations, scalability and high availability on sparse, distributed data. In general, NoSQL database is used in distributed system to reduce the pressure of main server [4]. The main difference in NoSql databases and relational databases is, NoSql databases do not need to design schema in advance. It stores the big data in order of rowkey.

Hbase is a NoSQL database stores data on disk in column oriented format. The basic data unit in Hbase is cell which includes the row id, column family name, column name and version or timestamp. Each Hbase cell can have multiple versions of particular data. Hbase uses Hadoop File System (HDFS) as its underlying data storage. At physical level, each column family is stored contiguously on disk and data is physically sorted by row id, column name and version [3]. Hbase handles shifted load and failures gracefully and transparently to the clients.

Section II describes background knowledge of difficulty with relational database management system, NoSQL database and Hbase. Section III describes the Quad tree index structure layer and Hbase data storage layer model. Section IV shows the experimental evaluation of nearest neighbor query and range query comparison of RDBMS with Hbase. We conclude our work in section V.

2. Related Work

Hbase [10], Cassandra [11], MongoDB [12] are the examples of NoSQL databases. These NoSQL databases support various levels of indexing. Our system uses a quad tree data structure which partitions the space into 2^n subspaces in a systematic manner along all dimensions.

Hbase is an open-source database which works on top of Hadoop[13]. It is basically known as column-oriented database based on the implementation of Google’s BigTable [14]. Hbase database consist of set of tables. Each table consists of row, column family, cell and version. A cell contains a data and a rowkey and column family name uniquely identifies a cell. Each data in a cell have a timestamp [13].

A. Difficulty with Relational Database System

RDBMS played an integral role when designing and implementing business applications which requires storage of information. Such type of applications retains information about number of users, products, orders [3] etc. So we are using rdbms as storage backend providing a persistent layer over frontend application server. This works fine when the records are limited, but when we have to store large number of records then such a type of relational database system shows some weakness.

B. NoSQL Database

A NoSQL database uses the model for storage and retrieval of data other than the tabular relations specially used in RDBMS. These databases use approach which is simple in design, horizontal scaling and high availability. For making some operations, NoSQL databases uses key–value pair which is differ in case of relational databases. NoSQL system also called “NOT only SQL” which may support SQL-like query operations.

NoSQL databases are highly used in big data problems and real time web applications where we want result in less time. Such types of databases designed with low cost commodity hardware and are easily scalable [7]. These databases use simple data models which lead to very less administration requirements and tuning requirements. Such databases use cheap commodity servers to manage large amounts of data and transaction data as compared to relational databases which uses expensive dedicated servers and storage systems. Hence, cost of NoSQL databases can be many times less than cost of RDBMS which allows you to store more amounts of data at much lower price.

NoSQL key value stores and document databases allow the applications to store any structures it wants in NoSQL databases. Open source implementation of Bigtable based NoSQL databases such as Hbase typically allows new columns to be creates easily. So the result is that if suppose application changes and database schema changes can be managed easily.

C. HBASE

Hbase is open source implementation of Google’s Bigtable. Hbase was created in 2007 at Powerset and initially it was a contribution part of Hadoop [2]. Then at later it has become top level project under the Apache Software Foundation umbrella. The latest release of Hbase version is 1.0.0.

Hbase is a distributed, persistent, strictly consistent and excellent read performance storage system. It efficiently uses disk space depending on the nature of data in column families. Hbase uses a single index rowkey, similar to primary key in RDBMS and It can offer the sever side hooks to implement the flexible secondary indexing. Hbase also provides filters for reducing data transferred over the network. There is no support for query language and it has limited support for transactions.

Hbase transparently handles the shifted load and failures. Hbase is scalable and clusters can grow and shrunk automatically and no need of complicated rebalancing or re-sharding procedure.

3. Quad Tree Index Structure on Hbase

The location data are inherently multidimensional consisting of spatial attributes (e.g. longitude and latitude). Different applications used such spatial data in various different ways. We develop a Quad tree index structure on top of Hbase which will provides real time query processing on given location.

A. Quad Tree Index Structure layer

We show how standard index structure like Quad tree [5] will be implemented on top of Hbase. The data storage layer, Hbase uniquely identifies the row by their rowkey. In the system the space is partitioned into different unique subspaces and each subspace stores the points equal to the physical bucket size.

For splitting the space, we used trie based quad tree. The bucket capacity must be fixed in the subspace, if the bucket capacity exceeds the specified limit then the subspace split into equal size subspace.

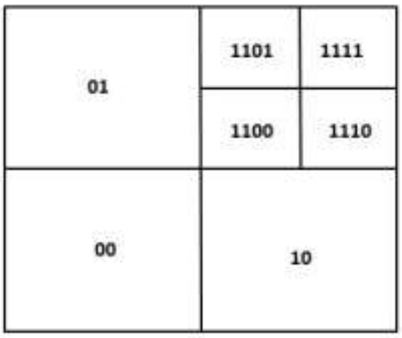

Figure 1: Quad Tree Index Structure

Trie-based quad tree has the property of z-ordering which states that, when the split occurs in space then all values in subspaces is in continuous order [1]. When multidimensional space is divided by using quad tree approach then we get four equal size subspaces. The z-order of given subspaces is calculated by interleaving the bits of subspaces.

Fig.1 shows, how space partition is done in trie based quad tree data structure. For example, before the space partition consists of z-value in z-order is 00,01,10,11. If we partition the space which consist of z-value 11, then after partition it will contain the z-value in z-order is 1100,1101,1110,1111. As the data grows and the capacity of the subspaces exceeds then subspace is divided into n-dimensional subspaces along all dimension resulting in 2n subspaces. Trie based quad tree splits the subspace into equal size and it is very efficient because it divides the space in regular shaped [1].

B. Hbase Data Storage Layer

We use Hbase as a data storage layer consists of range partitioned key-value store. It is open source implementation of BigTabel.

An Hbase table consists of number of regions, when data grows then Hbase automatically splits the data into different regions according to rowkeyandstores the data points of a particular mapping range [1]. The architecture of our system consists of two levels. The upper level is use for storing indexes and the lower level is responsible for storage of actual data points.

An Hbase installation in a system consists of collection region servers. If a data in regions exceeds beyond the configurable set limit then the Hbase splits the regions into two regions automatically. This states that Hbase is scalable and can handle growing amounts of data dynamically [1].

Our experiment uses all data points in a single table. For storing z-values for indexing of quad tree data structure we used a separate table.

The data points in a table are continuously stored in subspaces. When we have to search a data point then the z-value is calculated and then the actual table value is being fetched by rowkey. While inserting a data point, the corresponding entry in index table is being modified and then the actual data point is inserted into a table. When the space splits occurs then it updates the corresponding entry in the index table.

4. Experimental Evaluation

We implemented our work using Hbase 0.98.5. The machines run on 64 bit processor having 8GB RAM and 160 GB Hard Disk on Ubuntu 14.04 LTS. We used Eclipse as IDE for performing comparison analysis of RDBMS and Hbase. We used LAMP for RDBMS data storage platform. The evaluation consists of two standard queries processing, that is, nearest neighbor query and range Query over same data sets. We applied the quad tree multidimensional indexing on Hbase and spatial multidimensional indexing on RDBMS data storage.

We used data set with sufficiently large data points and organize these data points in space and check the performance of data organizations based on indexing technique applied on Hbase storage model and RDBMS storage model. The data points consist of mainly spatial data (i.e. latitude and longitude), userid, location, timestamp etc.

A. Nearest Neighbor Query Evaluation

Here we compare the Nearest neighbor query (KNN) performance using the same dataset on two data storage model, that is, RDBMS and Hbase. *Fig. 2* shows the result of comparison analysis of nearest neighbor query between RDBMS and Hbase. The bar chart shows that Hbase requires less response time as compared to RDBMS.

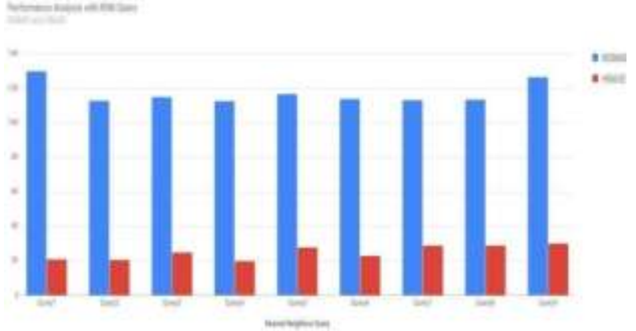

Figure 2: Response time of KNN query processing in RDBMS and Hbase

B. Range Query Evaluation

We compare the performance of Range Query on the same dataset using two data storage model, that is, RDBMS and Hbase. *Fig. 3* shows the result of comparison analysis of range query. The Bar chart shows that Hbase requires less response time as compared to RDBMS.

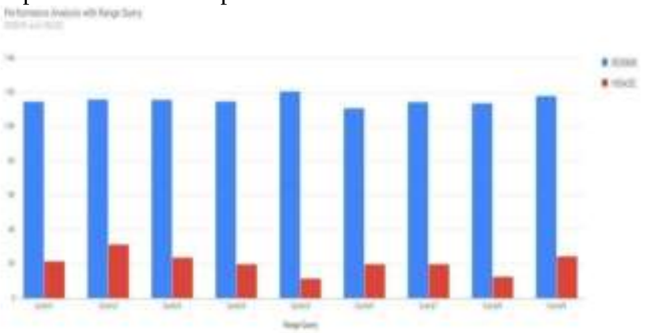

Figure 3: Response time of Range query processing in RDBMS and Hbase

5. Conclusion

We proposed Hbase as efficient and scalable data storage system which can supports efficient multidimensional nearest neighbor query and range query with response time less than the traditional relational data management system. We implemented our design using standard quad tree index layer and Hbase, open source key-value store. Our evaluation compares the multidimensional nearest neighbor query and range query in RDBMS and Hbase and shows that response time of Hbase is less than RDBMS.

References

[1] S. Nishimura, S. Das, D. Agrawal, and A. Abbadi, “Md-hbase: A scalable multi-dimensional data infrastructure for location aware services,” in Mobile Data Management (MDM), 2011 12th IEEE International Conference on, vol. 1. IEEE, 2011.

[2] HBaseThe Definitive Guide, 1st ed., O’Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

[3] A. S. Foundation, “Apache HBase Reference Guide,” April 2012. [Online].Available:http://hbase.apache.org/book/book.html

[4] J. Ernst, “SQL Databases v. NoSQL Databases,” Communications of the ACM, vol. 53(4), 2010, pp. 10-11

[5] R. Finkel and J. Bentley, “Quad trees a data structure for retrieval on composite keys,” Actainformatica, vol. 4, no. 1, pp. 1–9, 1974.

[6] Wikipedia, “Location-based service,” dec 2014.[Online].Available:http://en.wikipedia.org/wiki/Location-based_service

[7] R. Cattell, “Scalable sql and nosql data stores,” ACM SIG-MOD Record, vol. 39, no. 4, pp. 12–27, 2011.

[8] "Foundations of Location Based Services", Stefan Steiniger, Moritz Neun and Alistair Edwardes, University of Zurich

[9] Permanent Reference Document SE.23: Location Based Services“, GSM Association

[10] Apache HBase. http://hbase.apache.org/

[11] DataStax. http://www.datastax.com/

[12] MongoDB. http://www.mongodb.org/

[13] N. Dimiduk, A. Khurana, and M. H. Ryan, HBase in action, Shelter Island, NY: Manning, 2012.

[14] F. Chang, Dean, S. Ghemawat et aI., "Bigtable: A distributed storage system for structured data," Seventh Symposium on Operating System Design and Implementation, Seattle, WA: Usenix Association, 2006.

[15] Dan Han, EleniStroulia,"HGrid: A Data Model for Large Geospatial Data Sets in Hbase," in Cloud Computing, 2013 6th IEEE International Conference on, IEEE, 2013.