BANIAN: COLLABORATIVE QUERY SYSTEM FOR STRUCTURED BIG DATA

Bhavani V[1], Meena S[2], Menaka Rani R[3], Bindhu Madhavi P[4]

[1,2,3]*BE, Computer Science & Engineering, T. John Institute of Technology, Bengaluru, Karnataka, India*
preethiramm29@gmail.com
[4]*Associate Professor, Department of CSE, T. John Institute of Technology, Bengaluru, Karnataka, India*

Abstract

Increased development in structural data created new technological challenges in department of research in big data and relational database. In recent survey the submission of efficient system in maintaining and analyzing PB level structured data is otherwise said to be Banian. Storage structure limits in relational database has been crossed higher level compared storage structure. It effectively integrates interactive query was in a higher scale in the storage management . It has a same query interface to the cross platform dataset . It has two importance such as compatibility and scalability . The important stages in the Banian system architecture are: (1) Storage layer with help of HDFS for the massive data where storage is distributed; (2) for parallel database using splitting and scheduling technology we can perform scheduling and execution layer; (3) standard SQL provides cross platform query interface for application layer. Query optimization to a higher range 30times improved in Banian. To provide and enhance the performance and parallel execution.

Keywords: *Big Data; Interactive Query; Relational Database; HDFS; Cross Platform.*

---***---

1. INTRODUCTION

The mostly concentrated field in big data includes academic and industry. Scientists has discovered new ideas about the system and technologies which gives us a lot of gaining knowledge and information for the massive amount of data to be analyzed. GFS and Map reduce could progress 20PB of web pages per day in 2007 which is developed by Google. The HDFS and Hbase cluster developed by Facebook scanned 300 million images everyday in 2012, which is about more than 500TB of data. Baidu developed a search engine system which could handle 100PB of data daily in 2013.

The data has higher growth in coupling degree and high relevance which is improved and it has became famous in ecommerce and social network. Which showed the increased growth in the scale of structured data and PB level. The interactive query which was given more preference in the bulk volumes in data necessities, which showed the results in following management ability, rapid analysis and calculation capability requirement of which exposed .

Greater comparison for relational database and big data processing technology, one of the famous and well knowing database system of structured data and relational databases are Oracle, MySQL, SQL Server and DB2. In 1970 discovery of Codd was in the design of relationship, the further step was proceeded on relational method and theory of database. The development which crossed over the decades of relational database became one of the most popular technique used by the information management system and business application system.

In the data warehouse, it became a tool in analysis and it is one of the most accurate and high efficient for storage. The alternations happened in informational technology and architectural of relational database which showed a drastically development regularly, by the centralized database, distributed database to produce parallel database. The level in storage capacity which was improved from GB level to TB level in current days. Massive parallel processing based on parallel database , where data of hundred TB which can be managed by an architecture. The loosely coupled processing units are present in database system. Each and every unit consist of private computing and storage resources such as CPU, cache, memory, hard disk, operating system. Massive Parallel Processing database has its own specific features which are shared nothing and multiple copies of data. For the better performance we use different processing units which are executed concurrently and it can be split and scheduled. In PB level data they deal with the transaction mechanism and construction for index in relational database in which system expenses.

Google proposed a framework in programming which is called map reduce .It has typical technology and it goes with processing of big data. Processing big data has a scheduling capacity and high throughput with its super large scale management ability. Massive constructed data was well developed and it was performed by map reduce. Map reduce framework starts a job in each and every computing request. Map reduce framework helps to complete the job by performing two types of tasks map and reduce. The map and reduce which starts splitting of input database into independent blocks and it starts distributing to different nodes. Framework which is going to manage monitoring , scheduling the task and how to restart the failed task. The installation will be done at same time for computing nodes and storage nodes, where the task goes on in the node of the distributed file system. Framework which works to schedule looks on nodes with data storage effectively where the cluster network bandwidth will be completely used. In structured datasets and interactive query will not get support by the framework in map reduce. The initial step of the map and reduce task which takes its own time in resources system. The communication among sub-operation will only provide to intermediate files, where the complex commands should be decomposed. It takes more time for processing and it is intensified in higher level. The big data relational database deals with the study of structured big data. The splitting and scheduling model combines with HDFS where the interactive query and analysis will be integrated by large scale storage management by Banian.

2. SYSTEM ARCHITECTURE

The managing and analyzing of PB level structured data is designed in practical system which is efficient. The core component of Banian consist of scheduling and execution layer. It contains three modules scheduler, query engine, metadata server. The commands by SQL in the application layer will be received by scheduler. With HDFS Namenode the file information will be analyzed in to the position information of the data blocks. Operation list produced by scheduler for local execution. Metadata server stores metadata relevant to system to system. Metadata server will notify the scheduler about the files needed to be queried and the method of parsing the file content through the table structure. Metadata manages the fast lookup table for caching data block information. Query engine which deals in receiving and executing the operation list which was organized by scheduler and it is deployed on every sub node. During execution intermediate results are stored in memory. Once the operation list is completed query forwards its final results to the worker node. ETL offers various multidimensional choices to support upper application in terms of ETL cost, storage efficiency, and analysis performance.

The managing and analyzing of PB level structured data is designed in practical system which is efficient . The core component of Banian consist of scheduling and execution layer. It contains three modules scheduler, query engine, metadata server. The commands by SQL in the application layer will be received by scheduler . With HDFS Namenode the file information will be analyzed in to the position information of the data blocks. Operation list produced by scheduler for local execution. Metadata server stores metadata relevant to system to system. Metadata server will notify the scheduler about the files needed to be queried and the method of parsing the file content through the table structure. Metadata manages the fast lookup table for caching data block information. Query engine which deals in receiving and executing the operation list which was organized by scheduler and it is deployed on every sub node. During execution intermediate results are stored in memory. Once the operation list is completed query forwards its final results to the worker node. ETL offers various multidimensional choices to support upper application in terms of ETL cost, storage efficiency, and analysis performance.

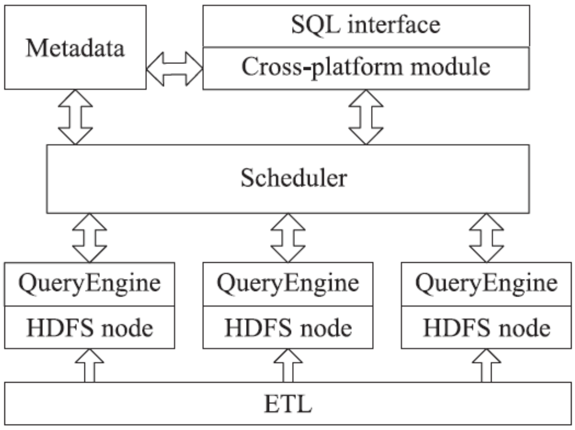

Fig. 3.1- Banian Architecture

3. SPLITTING & SCHEDULING

For achieving concurrent execution and for improving performance of the query system, the scheduling and execution layer will split SQL commands to subtasks and schedule those subtasks to different sub-nodes.

Below given figure shows how the scheduling and execution layer will process the SQL commands given by the user.

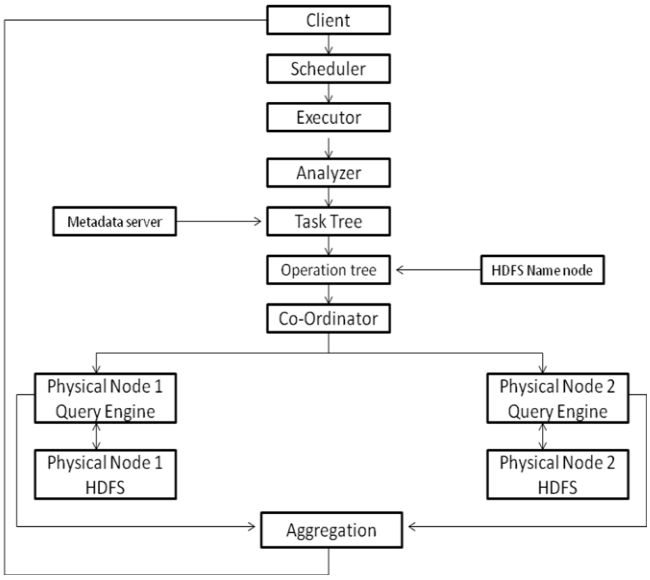

Fig. 2 Workflow of scheduling and execution layer

1) After receiving SQL commands syntactic and lingual analysis is done by the execution and analysis units to convert to task tree.

2) After task tree formation, traversing the task tree completely, obtain the metadata server according to table information, and then obtain the corresponding file information from HDFS name node.

3) Convert task tree in to operation tree, check the fast lookup table present in metadata server, if it is a cache hit then provide all these information to query engine present in each HDFS mode.

4) HDFS name node contain file block information, when this node gets file information it finds the correct data block information is fetched after traversing the operation tree completely.

5) After getting the data block information the operation tree of all sub nodes are integrated to form a operation list. The co-ordinator unit will send the operation list for the query engine to the corresponding sub node.

6) In step 4 and 5 make sure that all the data objects needed for operation are made available in local storage. Once the query engine receives the operation list it starts the execution by reading the data from local storage.

7) Once the execution of all commands in the operation list is completed there the results will be sent to the aggregation unit.

8) Aggregation unit will receive the result from all the nodes aggregate and send them back to the application layer.

In the below given figure 3 the scheduler is a logical unit. Each physical node is made with scheduler daemon. Here the scheduler is not a central node, and all the physical nodes will have the equal status. If user give any SQL command any scheduler can receive it and can act as a worker node to do the functionalities of scheduling layer. This system architecture is good at scalability and reliability. Because there is no central node for processing the SQL command. In this system any node can become central node and all the nodes have equal priority. In this it is possible to extend the cluster linearly and infinitely. The ability to query the commands will depend on the increasing size of the cluster.

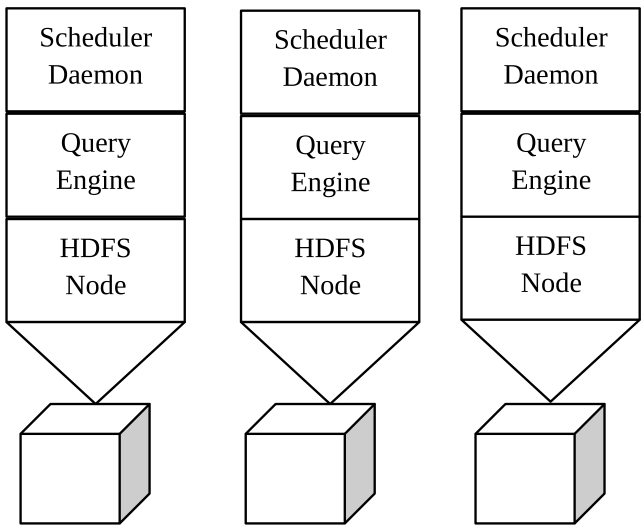

Fig 3. Scheduling Classification

As this system maintains multiple workers nodes it executes queries concurrently. This system architecture is efficient because it reduces the response time of the query by increasing the cluster nodes.

In a large scale distributed system break down detection is very much important scheduler is responsible for checking all the conditions and tasks to get completed in all the nodes. Suppose if there is a hardware failure, Network error, software failure, or any other failure, the scheduler will inform all the nodes about the failure node to avoid the execution in failure node.

4. CROSS-PLATFORM QUERY

In big data application data is stored in different platforms. In structured data for data extraction and loading the time cost cannot meet the real time requirement. As shown in fig.4 the different platform are interconnected using LAN or Internet which is known as distributed and heterogeneous Network topology.

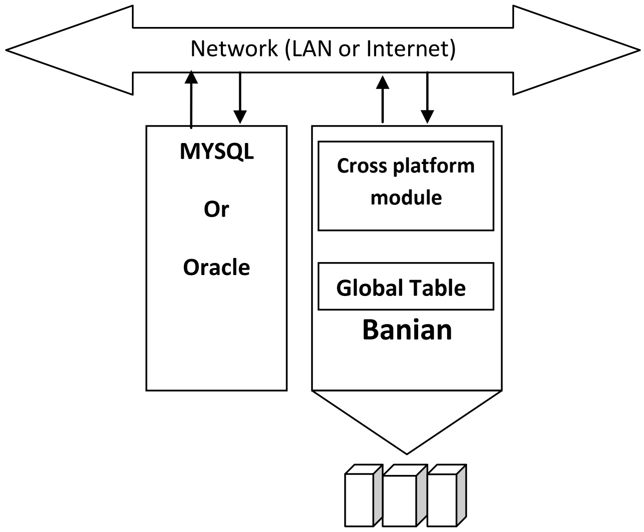

Fig 4. Cross Platform Network Topology

We make use of cross-platform query interface by which all the clients can execute their query between Banian or between Banian and any other relational database. The cross-Platform query interface has three components.SQL Interface Cross-interface module, and global table.

Through SQL interface user can give comments which will be sent to the cross-platform module. If the command provided by the user involves many data sets then the cross-platform module interacts with the global table to get the information of the location. After getting location information the commands are split according to the variable tag name of location, send the sub-command to slave platform and it will receive the result.

The global table stores the configurations information of all platform using a data structure called location.

```
Struct Location {
        Char    * tagname;
```

```
Char   * host;
Int      port;
Int      authority;
Char    *username;
Char   *pswd;
}
```

We will see the workflow of query execution by taking a join query. We will use two Banian systems called banian1 and banian2 with following values of Location structure,

```
banian1_Location {
        tagname='banian1';
        host='166.111.134.49';
        port=2276;
        authority=1;
        }

Banian2_Location {
        tagname='banian2';
        host='166.111.134.50';
        port=2276;
        authority=1;
        }
```

We will create database db1 on banian1 and add weblog table. We also create database db2 on banian2 and add table userinfo. Below given table shows how Cross-platform module will split SQL commands into sub-commands as C1 and C2.

Table 1. Cross-platform query command.

SQL command	Select * from banian1db1.weblog join banian2 db2.userinfo on banian1 db1.weblog.sourceip = banian2 db2.userinfo.sourceip where banian1 db1.weblog.time > 1401552000 and banian2 db2.userinfo.zipcode = 100084.
C1	Select banian2 db2.userinfo.sourceip from banian2 db2.userinfo where banian2 db2.userinfo.zipcode = 100084.
C2	Select * from banian1 db1.weblog where banian1 db1.weblog.sourceip = fresultg and banian1 db1.weblog.time >1401552000.

5. RESULT

We check the performance and scalability of Banian and compare the results with Hive. Banian is 2-7 times quicker than Hive. Banian is very good performer especially for complex queries involving multiple tables. The parallel splitting and scheduling technique is more advantage compared with MapReduce. If we increase the number of nodes, Banian is capable of handling query execution in parallel with same performance when compared with Hive.

6. CONCLUSION

In this system architecture it is capable of storing PB level data and perform interactive cross-platform query. In this architecture we implement HDFS with splitting and scheduling engine of parallel database.

The performance of of Banian is 5-30 times better than that of Hive when we use complex queries and multiple tables. For simple SQL statements the performance of Banian is 3-10 times better than Hive.

7. FUTURE SCOPE

To achieve better performance and scalability, Banian won't support the partial update and deletion of data table. We have to implement this update and deletion operations in this Banian architecture. After implementing above weaknesses this Banian architecture becomes full-fledged system.

REFERENCES

[1] Greenplum Inc., Greenplum Database: Powering the datadriven enterprise, http://www.greenplum.com/resources, 2014.

[2] Y. Xu, P. Kostamaa, and L. Gao, Integrating hadoop and parallel DBMs, in Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data, 2010, pp. 969–974.

[3] F. N. Afrati and J. D. Ullman, Optimizing Multiway joins in a map-reduce environment, IEEE Transactions on Knowledge & Data Engineering, vol. 23, no 9, pp. 1282–1298, 2011.

[4] H. Herodotou and S. Babu, Profiling, what- if analysis, and cost-based optimization of MapReduce programs, in PVLDB, 2011, pp. 1111–1122.

[5] Abouzeid, K. Bajda-Pawlikowski, D. Abadi, A. Silberschatz, and A. Rasin, HadoopDB: An architectural hybrid of MapReduce and DBMS technologies for analytical workloads, in PVLDB, 2009, pp. 922–933.

[6] A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka, N. Zhang, S. Antony, H. Liu, and R. Murthy, Hive — A petabyte scale data warehouse using Hadoop, in IEEE 29th international Conference on Data Engineering (ICDE), 2010, pp. 996–1005.