International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 06 Issue: 02 | Feb - 2022 ISSN: 2582-3930

Performance Requirements for Execution of Transactions in Distributed Database Systems

[1]Dr.T.A. Chavan, [2]Dr.D.B.Lokhande
[1]Principal, Shri Siddheshwar Women's College of Engineering, Solapur
[2]Assistant Professor, Shri Siddheshwar Women's College of Engineering, Solapur

Abstract – Accessing large databases from remote locations means dealing with different types of partitions and heterogeneous machines. The advantage of replication is that if faster access of database is required it must be stored on local machines. To synchronize and update data in all the replicas is not an easy task. To accomplish such task, various methods and techniques are available. In this paper, the requirements related to performance of transaction management in distributed database management system are taken into consideration.

Index - Distributed Database Systems, Partition, Performance requirements, Replication, transaction management.

I.INTRODUCTION:-

Distributed Database means the database is partitioned or fragmented and stored on different heterogeneous machines. Through transaction management system such database is accessed by various kinds of applications which is stored in fragments on different types of machines. There are various types of applications which uses this database simultaneously and may resulted into a conflict.

The database management system plays important role to solve such conflicting situations. In Concurrent execution the transactions are executed simultaneously by DBMS. Simultaneous execution of transactions may result into inconsistency if proper concurrency control mechanism and integration of data is not taken into account.

To successfully execute transactions in distributed database systems distributed processing is required but it is very difficult task to manage. Heterogeneous distributed database means it is the database which is distributed and stored in different types of machines and different types of database management systems.

To access data faster from database fragments are replicated in distributed database management systems.

Because database is scattered on different locations and on different kinds of machines performance requirements needs to be considered and resolved to better execute the transactions. Such performance requirements are discussed in this paper.

II.PRINCIPLES OF DDBMS:-

The principles of Distributed Database are as follows:-

1.Partition [1] :-

Partition is the concept though which the data is distributed across multiple computers. To divide the data across the distributed systems, there are different types of partitioning techniques are available.

The most important techniques of partitioning are horizontal partitioning, vertical partitioning and hybrid partitioning. Fragmentation is also another name of partion.

2.Replication[1] :-

For fast access the fragmented data are duplicated and stored on multiple computers.

Among all the replicas Synchronization is required which is required to access correct data.

3.Transparency [1]:-

User should be able write and execute a query without giving any attention to data distribution, data location fragmentation / partition or replication.

This kind authorization to users is called Full Transparent access. If database is of the distributed nature then full access transparency is needed at different levels of distribution.

Full access transparency is required in the form of Data independence, Network transparency, Replication transparency and Fragmentation transparency.

4.Distributed Query Processing [1]:-

Distributed Query processing deals with optimization. It includes working on algorithms which analyze queries and converts them into a series of data manipulation operations.

5. Distributed Concurrency Control [1] :-

Database integrity is the most important task while performing concurrent operation on the database.

Distributed Concurrency control includes the synchronization of data accesses so that the integrity of the database is achieved.

6.Replication Protocols[1]:-

It is important to implement protocols which ensures the consistency of the replicas when the distributed database is replicated partially or fully.

7.Heterogeneity[1]:-

The word 'Heterogeneity' could occur in different forms in distributed systems such as DBMS ,hardware, network, , Data models, query languages, transaction management protocols, etc.

Data Representation with different data modeling tools and techniques invites heterogeneity because of the limitations and restrictions of individual data models.

III. DISCUSSION OF DDBMS ISSUES:-

1.Synchronization among replica[1] :-

For accessing data quicker and faster, the database fragments are replicated on different machines. It is very difficult to synchronize the replicas because when data on any one replica is updated, the changes should also be reflected on other replica to provide correct information.

2. ACID Properties :- Today, most of the applications are distributed by nature such E-commerce application, Web-based applications, multimedia applications such as medical imaging, etc. In RDBMS i.e. Relation Database Management Systems, the transactions must satisfy the ACID(Atomicity, Consistency, Isolation, Durability) properties[2].

But in distributed database management systems it is not possible to get all the four properties together. According to Eric Brewer [2010] has given the CAP theorem which shows that it is impossible for a distributed database management system to satisfy Consistency, Availability and Partition tolerance simultaneously. i.e., a distributed database management system cannot provide all three of these guarantees at the same time.

It is very difficult to maintain ACID properties across partitions because it is needed to restore partitions to ensure it.

3.Data stream management[1] :-

Fixed schema structure is not useful when dealing with data which comes in free structure from different kinds of machines over the internet. To deal with such types of streamed data, good stream data management is required.

4.Query Optimization[3] :-

To optimize the join queries which are distributed in nature is not an easy task.

IV.RELATED WORK AND FUTURE SCOPE:-

In [1] Discussed that there are algorithms available for fragmentation or partitioning. In web-based applications data comes in the form of audio, video, documents and other formats which are derived as 'stream data'. The Real- time data occurs in the form of stream i.e. unbounded sequence from internet.This data is distributed across many machines which is mostly accessed by different users from their own machines.

In [4] has elaborated that to optimize distributed join over databases with efficiently considering run-time conditions by using Cluster and Conquer algorithm. The motivation behind Cluster and Conquer algorithm is taken from real world observation in which author has viewed whole database federation as clustered system which in turn gives cluster mediator.

At the last the author has executed the prototype federation system with the proposed architecture and optimization algorithm. This experimental analysis and results proved the capabilities and efficiency of Cluster and Conquer algorithm which gave the target environment where the algorithm performs better than other related approaches.

At present the prototype system maintain two levels of mediators, but actually the requirement is to extend the system in order to support multi-level mediators whenever the environment need arises. One another important extension is to use this algorithm to other distributed systems, such as distributed databases and grid computing systems. The concept of cluster and conquer is anticipated to be useful for large-scale distributed computing environments. Due to this the algorithm can be extendedfor the processing of other types of operations such as aggregate i.e. group-by, max and min,top-K, etc.

In [3] proposed a cost based model which permits internal operator parallelism opportunities which needs to be identified within query execution plans. This makes the **response time** of a query to be evaluated more accurately. In [3] has combined two existing centralized optimization algorithms to make a more efficient and productive algorithm. The author has elaborated a novel multilevel optimization algorithm framework which combines heuristics with existing centralized optimization algorithms.

In this paper the distributed multilevel optimization algorithm (DistML) introduced which utilizes the concept of distributing the optimization phase among the multiple optimization sites in order to fully utilize the available systemresources.

The future work on such cost based model can be extended to make it capable for handling pipelining between operators that means a operator can feeds its own output tuples exactly into a parent operator when they become available without writing them to disk.

In [5] has studied the join techniques on a 100-node system and proved the unique tradeoffs of these join algorithms in the context of MapReduce algorithm. In [5] explained how the join algorithms can be valuable from certain types of practical preprocessing techniques.

The beneficial insights gathered from their

study can help an optimizer select the appropriate algorithm based on a few data and characteristics. The proposed techniques can be examined for multi-way joins, using indexing methods to speedup join queries, and designing an optimization method which can automatically select the appropriate join algorithms.

Another important direction about future is to create novel programming techniques to extend the MapReduce framework for more advanced techniques.

For Cost calculation from the optimization method, the optimizer should consider the data sources involved in an operation to find the costof that operation.

In [6] indicate that when the physical database design is already known to the optimizer, the given query optimizationalgorithm works very well. But in absence of physical database design, more effective optimization methods should be required.

In [7], a new algorithm is designed, and experiment results shows that the algorithm can mainly reduce the amount of intermediate result data, reduce the network communication cost, to increase the optimization efficiency. For the future work, the algorithm can be enhanced for distributed file system.

V.CONCLUSION: -

To find an ideal optimum solution for the distributed data is not an easy task. To obtain optimum solution the cost of network, resources, response time, access time, memory usage, processing time, etc. should be minimized which could be possible with the use of recent algorithms for distributed DBMS.

VI.REFERENCES:-

1. M. T. Ozsu, and P. Valduriez, “Book1 : Principles of Distributed Database Systems, Third Edition, Springer

2. S. Naik, “Book1-Concepts of Database Management Systems”, First Edition, Pearson

3. R. Ttaylor, “Thesis on Query Optimization for Distributed Database Systems”, University of Oxford, 2010-11

4. D. Wang, “Thesis on Efficient Query Optimization for Distributed Join in Database Federation”, Worcester Polytechnic Institute, March 2009-10

5. S. Blanas, J. M. Patel, V. Ercegovac, J. Rao, E. J. Shekita and Yv. Tian, “A Comparison of Join Algorithms for LogProcessing in MapReduce1”, SIGMOD’10, June 6–11, 2010, Indianapolis, Indiana, USA.

6. D. Sukheja and Uv. Singh, “A Novel Approach of Query Optimization for Distributed Database Systems”, IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 4, No 1, July, 2011-12

7. F. Yuanyuan and MM. Xifeng, “Distributed Database System Query Optimization Algorithm Research”, IEEE, 2010-2011.

BIOGRAPHIES:-

Dr.T.A.Chavan is the Principal of Shri Siddheshwar Women’s College of Engineering. He is having Ph.D. in Computer Science and Engineering. He has 24 years of teaching Experience.

Dr.Dheeraj Bhimrao Lokhande is Assistant Professor in Shri Siddheshwar women college of Engineering. He is having Ph.D.in Computer Science and Engineering. He has 12 years of teaching Experience.