CSCI 5408: Assignment 2

<u>Problem 1</u>:

Analysis of Joins and Semi-joins in Centralized and Distributed Database Queries [1].

The above research paper makes a comparison of Semi-Joins and Joins operations in a Centralized and Distributed Database System by computing and analyzing their performance based on various factors like Query Cost, Memory used, CPU Cost, Input-Output Cost, Sorting Operations, Data Transmission, Total Time and also the Response Time. The focus is on comparing and contrasting between Joins and Semi-Joins. A complete in-depth experimental analysis is performed on centralized and distributed systems using both the joins under various circumstances. Tables and Figures are also used to draw a visual comparison of these joins and conclude which join would be better for a particular system.

Joins and Semi-Joins are some of the most fundamental and imperative operations in the database used to extract data from multiple tables at once. The Join of Equi-Join performs joining by matching the tuples based on some attribute that is being passed. On the other hand, Semi-Join reduces the size of the relation used as an operand, reducing the data exchange quantity. The trade-off for reducing data exchange is increased local processing cost and the number of messages being received.

An experimental analysis is performed on both the joins in a Distributed and Centralized Database systems. The metric analysis considers the following metrics Cost, Bytes, CPU Cost, and I.O. Cost. Based on the readings recorded, the Simple Join operation performed exceptionally well compared to the Semi-Join, majorly on the Select and Nested Loops statement. Bar chart and Pie chart are used to map the experimental readings for better visualization.

A similar analysis is done for the joins on a Distributed Database system, where the measuring factors are Equivalent relational algebra's operations, Placement of data and application programs, Bytes transferred, Total time and Response time. The results of this analysis showed that Semi-join performed much better as compared to the Simple Join. Finally, another analysis is done for different query plans based on Plan Cost, Logical Reads, Scan Rows, Sort Rows, and Memory. Again, data suggests that the query shows significant change when executed with different query plans.

The research paper details various scenarios where a query optimizer is being used. The query Optimizer tool is an essential tool used to find alternative suggestions for queries highlighting the reason behind the efficiency of queries.

The research paper finally concludes that the efficiency of Semi-Joins is more in Distributed Database whereas Simple Joins perform well in Centralized Database. The reasons are for Semi-Joins less data is being fetched, making it more efficient and fast, whereas for Simple Joins, all data is being fetched, leading to higher processing times. On the other hand, in Centralized Database, Simple Joins use up less CPU power, whereas Semi-Joins require more processing power. Therefore, it is difficult to conclude which one is better. However, it is evident that Semi-Joins transmit lesser data and implement more operation on comparison as compared to the Simple Join.

A Survey on Distributed Deadlock and Distributed Algorithms to Detect and Resolve Deadlock [2].

The primary focus of the research paper is on Operating System (O.S.) and Distributed Database System (DDS) being the most susceptible to deadlocks. DDS is basically a database system that stores multiple logically connected databases on a distributed computer system connected over an extensive network to achieve optimal performance and sharing of resources. The paper states what causes a deadlock, what techniques are used for deadlock avoidance, its prevention and ways to recover.

The paper gives an example of various university colleges in different cities for Distributed Database System (DDBS). DDBS uses concurrency control techniques to maintain a database's consistency when simultaneous transactions are being executed. According to the paper, one of the ways that deadlock occurs is through transaction locks. For example, when a transaction locks data A, and the second transaction locks data B, and if the first transaction tries to access data B and the second tries to access data A, then a deadlock is formed.

To recover from a deadlock, data structures like Wait For Graph (WFG), Probes, and Tables. The paper proposes various algorithms that could be used simultaneously to detect and avoid deadlocks. Deadlock basically represents a cyclic structure that can be detected using a directed graph similar to WFG. Another way is to use messages, so Probes follows the messaging way of notifying the transactions stuck in a deadlock to finish or release the lock on resources, thus releasing the deadlock. While using probes, whenever the message comes back to the initial transaction being notified that initiated the probe, a deadlock is confirmed again, leading to a cyclic structure.

Deadlock detection requires two tables being used by B.M. Alom Algorithm along with WFG. The working is quite simple, one of the tables of the B.M. Alom Algorithm maintains all those transactions that are present in a deadlock. The second table keeps the data regarding which transaction is most important, i.e. priority matrix of the transactions of a deadlock. The very first transaction being suspended is the least important one in a deadlock. One factor that lowers the efficiency of the B.M. Alom Algorithm is continuous changing of transaction importance leading to lowered throughput and resolution. Another algorithm that is somewhat in the similar lines of the B.M. Alom Algorithm is the Edge-Chasing Algorithm. The point of failure for this algorithm is that if the deadlock initiating the transaction is not present in the deadlock.

The paper's conclusion covers the shortcomings of both the discussed algorithms and discusses other techniques like time stamping for aborted transactions. The paper uses visualizations explaining both the algorithms but no technical details are mentioned causing a gap in understanding the algorithms and their impacts.

References:

[1] M. Sharma and G. Singh, "Analysis of Joins and Semi-joins in Centralized and Distributed Database Queries," 2012 International Conference on Computing Sciences, Phagwara, 2012, pp. 15-20, doi: 10.1109/ICCS.2012.15.

[2] V. Kate, A. Jaiswal and A. Gehlot, "A survey on distributed deadlock and distributed algorithms to detect and resolve deadlock," 2016 Symposium on Colossal Data Analysis and Networking (CDAN), Indore, 2016, pp. 1-6, doi: 10.1109/CDAN.2016.7570873.