DALHOUSIE UNIVERSITY

CS 5408 – Data Management, Warehousing and Analytics

Assignment – 2

Problem 1

Prashit Patel - B00896717

- **Reading Material 1: Analysis of Joins and Semi-joins in Centralized and Distributed Database Queries – Summary and Analysis**

  The purpose of the paper by Manik Sharma is to understand and analyze different approaches of executing a query using joins and semi joins in centralized and distributed database systems based on different metrics.

  Data is located at a single location in a centralized database system whereas data is divided and sometimes replicated at different physical locations in a distributed database system. Data extraction is easy from centralized database system than its counterpart. Also, extra effort is needed to decide placement of data and transforming a query to lower-level equivalent queries.

  Joins are used to extract information form more than one database tables. Comparison is made between joins and semi joins. Semi joins optimize the join query as it reduces the data exchanged but at the same time it increases local processing cost and number of messages passed.

  Metrics such as cost of query, bytes accessed, cardinality and optimizer used are considered for analysis. When comparison was made between the metrics for query using join and semi joins, it was found that for centralized database system, cost, IO cost and CPU cost was more for semi join than join while the bytes accessed was almost similar. These results contradicted the common notion that semi join helps in reducing the disk access.

  The same query was analyzed with Toad for Oracle software. The metrics used for comparison was namely logical reads, scanned rows, sort rows and memory. The query showed significant changes when compared with the best suggested query plans for the software. Significant change was found in logical reads and scan rows in one of the query plans compared to original query plan.

  Comparing the same for distributed database systems, metrics such as total cost and response cost were considered. Total cost was the addition of local processing cost and communication cost while response cost was the time elapsed from starting to completion of query. Three scenarios were considered. Copying both tables to result site, copying one table to another table's site and vice versa. The same query was then implemented with semi joins. The results showed that total time was reduced with semi join in distributed database systems. Also, data transmission was on the lower side in case of semi joins in distributed database systems.

  The central idea of the discussion is that join is better in performance compared to semi joins for centralized database systems. It is the same case except when data transmission is of utter importance.

  The topic which I found most interesting are the comparisons of joins and semi joins with different query plans considering different metrics. Also, actual calculations were shown for comparison of queries which made it easy to compare and understand the actual extent of differences.

- **Reading Material 2: Survey on Distributed Deadlock and Distributed Algorithms to Detect and Resolve Deadlock – Summary and Analysis**

Distributed Databases are those where a database in divided and stored among different physical locations. In distributed databases, transactions are executed concurrently which often results in deadlock. A deadlock is a situation where two or more processes are waiting for release of one or more resources.

Locking is used to maintain integrity of data. Following 2 rules are considered for locking:

- If any transaction is performing read operation, any number of other transactions can perform read operations on same data.
- If any transaction is performing write operation, no other transactions can access the data until the write operation is completed.

Based on these above rules, 2 types of locks are defined namely: Shared lock where data can be shared as mentioned in rule 1 and Exclusive lock where a resource is exclusively allocated to a transaction and no other transaction can access it as mentioned in rule 2 above.Few algorithms are discussed for deadlock detection:

1. Algorithm by Chandy – Uses transaction wait for graphs (TWFG). Probe computation is used where a transaction determines if it is in deadlock or not by sending a probe to transactions in local site. A transaction can issue maximum of one probe. If it receives a probe in response than it is considered in deadlock state.
2. Obermack proposed a similar type of algorithm for a external node but it detected false deadlocks.
3. Ho et al proposed an algorithm where a table was maintained for the resources held and a central site was responsible for deadlock detection.

Deadlocks are represented by a TWFG graph where transactions are represented by set of vertices and waiting states are represented by set of edges. If a cycle is present in the graph, it represents deadlock.Deadlock handling strategies:

1. Deadlock avoidance – It tries to avoid a deadlock by performing a check before hand for a global safe state but was not workable.
2. Deadlock Prevention – It assures that atleast 1 condition leading to deadlock must never hold but it was not workable too and provided inefficiencies.
3. Deadlock Detection and Recovery – It allows deadlock to occur, detects via WFG graph and waits till cycle is broken.

B.M. Alom Algorithm – This detects local as well as global deadlocks. Maintains 2 tables along with priorities. If deadlock occurs, transaction with lowest priority was aborted. Disadvantage was changing priorities.

Detection and Resolution of global deadlock – It identifies cycles and breaks one by aborting lowest priority transaction.

Edge chasing algorithm – It uses special messages, probes along the path to detect deadlock. If it arrives to transaction which initiated it then deadlock exists otherwise not.

The idea which I liked regarding this paper was it provides different methods of detecting as well as avoiding deadlocks.