ABSTRACT

Various open source database management systems such as Minirel, CUBRID, MariaDB,

SmallSQL, Minibase, etc. have been developed to aid in understanding of internals of a database management system. In this report, emphasis has been laid on Minibase which provides access to different components of a database management system such as the buffer manager, the storage structure, execution plan generator, query optimizer and the disk space management system.

Among the various operations that a database management system is responsible for, Joins turn out to be very essential operations. Among the various techniques that have been proposed for carrying out joins efficiently, not many of those optimization have been applicable to Joins with inequality predicates.

In this phase, we implement IEJoin algo as proposed by … et al and compare its performance with sort-merge join and nested loop join with inequality predicates. It is evident from results that IEJoin outperforms other techniques for both single and two predicate inequality joins.

Keywords - Nested Loop join, Sort-Merge join, IE Join.

INTRODUCTION

Joins are one of the most crucial operators in a database. Based on the join predicates and the algorithm used for the join, the result generation may range from a few seconds to a few days. Thus, careful analysis of the the join techniques is essential for implementing the most efficient algorithm for the relations and the predicates in consideration.

Minibase supports Nested loop join and Sort-Merge Join [1]. To optimize the performance of selections and projection on attributes, these operations are implemented while reading data from files. To access data from the files containing records, Iterators are used which can return data in desired sorted order.

Join is generally performed by doing a selecting or projection on a Cartesian product operation. This operation is time consuming and hence some databases use indices such as B+trees. Using B+trees may not result in full performance is the index is unclustered.

IE Self Join

As an extension to the existing algorithms, we implement the Space Efficient Inequality join ( IE Join) as described in ….. [Paper]. Two variants of the IE join exist: one for the generic two predicate join between two tables and the other one for two predicated self join. For join with the same table, we implement the algorithm using permutation arrays which map the occurrences of records in one sort order to the occurrences of those records in another sorted order. The permutation array is scanned sequentially and a track of the scanned records in maintained in a bit array which allows for space efficient calculation of the joins.

IF Join

For the generic two table IE Join, we create the permutation array for both the tables by proceeding in the same way as in case of two predicate IE self join. Two offset arrays are also created for both join attributes in the respective predicates. These offset arrays map the position of records sorted on X attribute in R table to the record sorted on X’ attribute in S table where X maps to X’. Finally, a bit array is maintained as in case of self join for computing the join tuples. The offset arrays along with the permutation arrays help in accessing the records according to constraints imposed by the specified predicates.

References

[1] http://research.cs.wisc.edu/coral/mini\_doc/joins/j.html