

Query Optimization in Centralized or Distributed Systems



Query Optimization in Centralized Systems

Query Optimization in Centralized Systems:

In a centralized system, query processing is done with the following aim – Minimization of response time of query (time taken to produce the results to user's query).

Maximize system throughput (the number of requests that are processed in a given amount of time).

Reduce the amount of memory and storage required for processing.

Increase parallelism.



Query Optimization in Centralized Systems

Query Parsing and Translation

Initially, the SQL query is scanned. Then it is parsed to look for syntactical errors and correctness of data types. If the query passes this step, the query is decomposed into smaller query blocks. Each block is then translated to equivalent relational algebra expression.

Steps for Query Optimization

Query optimization involves three steps, namely:

- **query tree generation**
- **plan generation**
- **query plan code generation**



Query Optimization in Centralized Systems

Step 1 – Query Tree Generation

A query tree is a tree data structure representing a relational algebra expression. The tables of the query are represented as leaf nodes. The relational algebra operations are represented as the internal nodes. The root represents the query as a whole.

During execution, an internal node is executed whenever its operand tables are available. The node is then replaced by the result table. This process continues for all internal nodes until the root node is executed and replaced by the result table.



Query Optimization in Centralized Systems

Step 2 – Query Plan Generation

After the query tree is generated, a query plan is made. A query plan is an extended query tree that includes access paths for all operations in the query tree. Access paths specify how the relational operations in the tree should be performed. For example, a selection operation can have an access path that gives details about the use of B+ tree index for selection. Besides, a query plan also states how the intermediate tables should be passed from one operator to the next, how temporary tables should be used and how operations should be pipelined/combined.



Query Optimization in Centralized Systems

Step 3– Code Generation

Code generation is the final step in query optimization. It is the executable form of the query, whose form depends upon the type of the underlying operating system. Once the query code is generated, the Execution Manager runs it and produces the results.



Approaches to Query Optimization

Exhaustive Search Optimization

In these techniques, for a query, all possible query plans are initially generated and then the best plan is selected. Though these techniques provide the best solution, it has an exponential time and space complexity owing to the large solution space. For example, dynamic programming technique.

Heuristic Based Optimization

Heuristic based optimization uses rule-based optimization approaches for query optimization. These algorithms have polynomial time and space complexity, which is lower than the exponential complexity of exhaustive search-based algorithms. However, these algorithms do not necessarily produce the best query plan.

.



Query Optimization in Distributed Systems

Distributed Query Optimization

Distributed query optimization requires evaluation of a large number of query trees each of which produce the required results of a query. This is primarily due to the presence of large amount of replicated and fragmented data. Hence, the target is to find an optimal solution instead of the best solution.

The main issues for distributed query optimization are :

- 1. Optimal utilization of resources in the distributed system.**
- 2. Query trading.**
- 3. Reduction of solution space of the query.**



Query Optimization in Distributed Systems

1. Optimal Utilization of Resources in the Distributed System:

A distributed system has a number of database servers in the various sites to perform the operations pertaining to a query. Following are the approaches for optimal resource utilization:

- **Operation Shipping**
- **Data Shipping**
- **Hybrid Shipping**



Query Optimization in Distributed Systems

Operation Shipping:

In operation shipping, the operation is run at the site where the data is stored and not at the client site. The results are then transferred to the client site. This is appropriate for operations where the operands are available at the same site. Example: Select and Project operations.

Data Shipping:

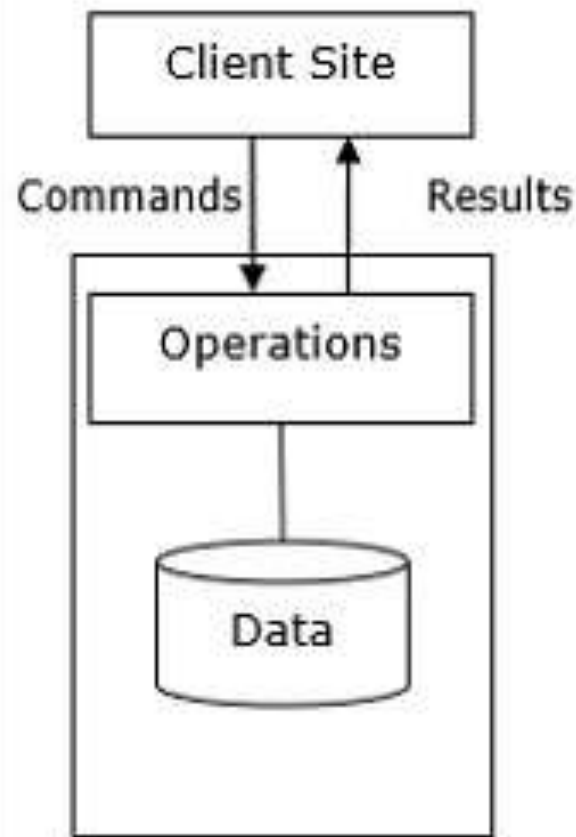
In data shipping, the data fragments are transferred to the database server, where the operations are executed. This is used in operations where the operands are distributed at different sites. This is also appropriate in systems where the communication costs are low, and local processors are much slower than the client server.

Hybrid Shipping:

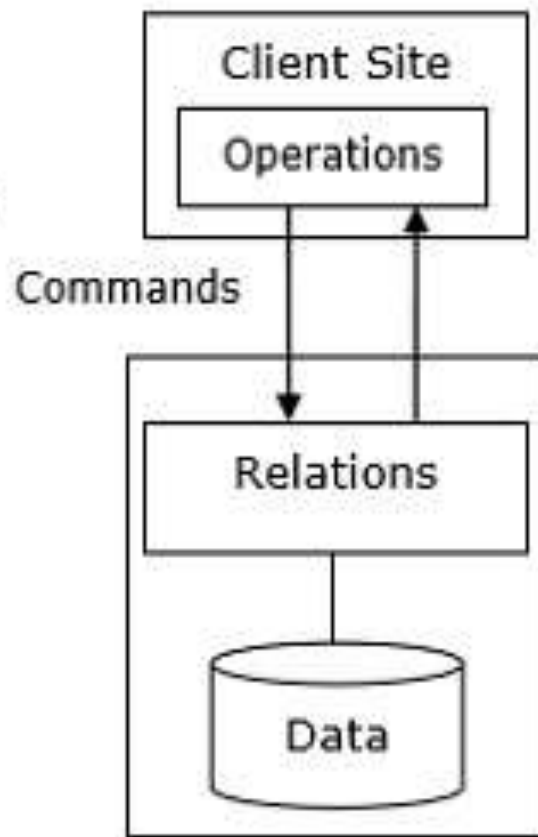
This is a combination of data and operation shipping. Here, data fragments are transferred to the high-speed processors, where the operation runs. The results are then sent to the client site.



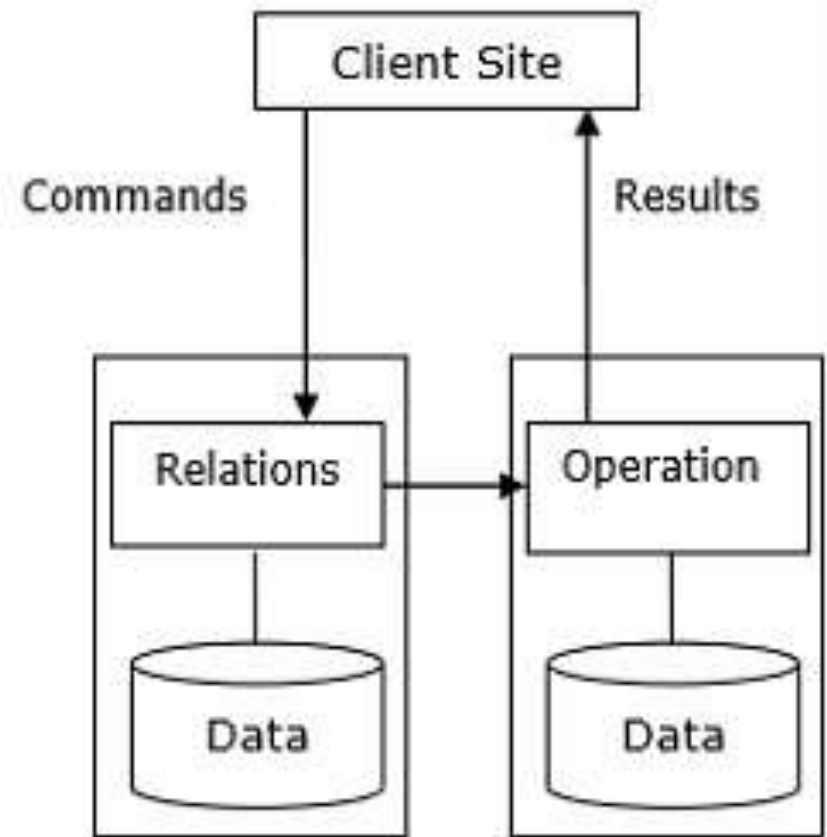
Query Optimization in Distributed Systems



Operation Shipping



Data Shipping



Hybrid Shipping

Query Optimization in Distributed Systems

2. Query Trading:

In query trading algorithm for distributed database systems, the controlling/client site for a distributed query is called the buyer and the sites where the local queries execute are called sellers. The buyer formulates a number of alternatives for choosing sellers and for reconstructing the global results. The target of the buyer is to achieve the optimal cost.

The algorithm starts with the buyer assigning sub-queries to the seller sites. The optimal plan is created from local optimized query plans proposed by the sellers combined with the communication cost for reconstructing the final result. Once the global optimal plan is formulated, the query is executed.



Query Optimization in Distributed Systems

3. Reduction of Solution Space of the Query:

Optimal solution generally involves reduction of solution space so that the cost of query and data transfer is reduced. This can be achieved through a set of heuristic rules, just as heuristics in centralized systems.

Following are some of the rules :

- Perform selection and projection operations as early as possible. This reduces the data flow over communication network.
- Simplify operations on horizontal fragments by eliminating selection conditions which are not relevant to a particular site.
- In case of join and union operations comprising of fragments located in multiple sites, transfer fragmented data to the site where most of the data is present and perform operation there.
- Use semi-join operation to qualify tuples that are to be joined. This reduces the amount of data transfer which in turn reduces communication cost.
- Merge the common leaves and sub-trees in a distributed query tree.



THANK YOU

ANY QUERY???

