Query Optimizer

The query optimizer optimizes the execution plan of a query.

When writing a GSQL query with multi-hop `SELECT` statements, the path pattern (Syntax V2) specified in the `FROM` clause is declarative. In other words, it specifies the structure of paths to be matched, without specifying the particulars of how the graph is traversed to find these paths. It is the query optimizer's job to determine the best graph traversal plan to satisfy the query.

Heuristics-based optimization

By default, the query optimizer chooses traversal plans based on heuristics. That is, it looks at the syntax of the query and makes a best-effort estimate to determine the traversal plan. This happens automatically during the installation of a query.

However, since heuristics-based optimization does not take the actual distribution of data into account, it sometimes would produce less-than-optimal traversal plans.

Cost-based optimization (Preview)

TigerGraph 3.7 adds the capability of performing cost-based optimization to the query optimizer.

In cost-based optimization, the query optimizer utilizes various pre-computed statistics describing the data distribution of the types referenced in the query. It then optimizes the execution plan to traverse the graph based on those statistics to minimize the cost of executing the query.

As of 3.7, the Preview version of cost-based optimization is focused on reducing query execution time.

To enable cost-based optimization, see Enabling Cost-Based Optimization.

For references on the REST endpoints for gathering statistics for the optimizer, see Statistics REST APIs.

Last updated 2022-09-29 09:21:20 -0700