714. Measuring Performance Impacts

In order to validate the optimizations described in the page on query optimization, several tests were carried out applying individual as well as combined measures. The methods and results are described on this page both to give an estimate of the performance benefits that can be achieved as well as allow other groups to repeat these measurements in their own setting.

Validation Environment

Validation was carried out using a large dataset (19GB CSV clinical raw data, resulting in 63GB after loading, including indexes). To eliminate side-effects due to caching mechanisms both of the underlying operating systems as well as the database, a virtualized environment (Oracle VirtualBox) was use on a standalone PC without concurrently running applications. Separate virtual machines (VMs) were created for the database (Oracle 11g) and the i2b2 application server. Snapshots were taken after loading, and after each measurement the VMs were returned to those snapshots before applying the next optimization method or combination thereof.

For each measurement, queries were manually constructed in the i2b2 Web Client, started, and the run time displayed by the application noted. For each query and optimization method, 2 separate measurements were carried out (returning to the snapshots in between), with the average of both measurements taken as the final value. The queries were constructed to reflect the impact of the optimization methods (e.g. one query combined rare and common data elements to show the effect of query strategy optimization, another query combined low- and high volume data elements to show the effect of partitioning; another query contained modifiers to show the effect of index optimization).

Validation Results

Query strategy optimization (through population of the C_TOTALNUM column) yielded performance improvements of 3x for one query, but led to unchanged or degraded performance in 3 other queries. This effect was surprising and may be related to the assumption that the data element frequencies measured on the overall dataset may no longer be accurate for consecutive query steps, which apply only to chosen subsets of the population.

Index optimization yielded performance benefits of up to 70x and did not lead to performance degradation in any of the queries. The modified indexes however took up 22GB of additional disk space.

The range-based partitioning yielded performance benefits of up to 80x in 3 queries, with degraded performance in one query.

Explicitly updating the table statistics did not lead to any change in performance across all queries. This result may be due to the default setup of the database used in the validation environment, which have led to up-to-date table statistics being present in the base snapshot. This issue was not further examined in our project.

The combination of all optimizations yielded performance gains of up to 78x across all queries. The slightly diminished yield compared to 80x with partitioning may be due to negative effects of the query strategy optimization.

Conclusions

Index optimization yielded the highest performance gains, which can be achieved with little change to the basic setup of i2b2: only one index needs to be replaced, and data does not need to be reloaded. Also, there is no enterprise or partitioning feature license required for index optimization. Partitioning may provide further performance gains with larger datasets, which was not verified in our project.