Don't Hold My Data Hostage - A Case For Client Protocol Redesign

Mark Raasveldt
Centrum Wiskunde & Informatica
Amsterdam, The Netherlands
m.raasveldt@cwi.nl

Hannes Mühleisen
Centrum Wiskunde & Informatica
Amsterdam, The Netherlands
hannes@cwi.nl

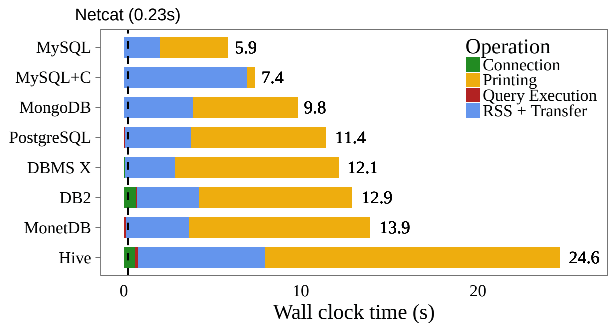

Figure 1: Wall clock time for retrieving and printing 1M `lineitem` entries from a local database. The dashed line is the wall clock time for netcat to transfer a CSV of the data.

ABSTRACT

Transferring a large amount of data out of a database system to a client program is a common task. Examples include complex statistical analysis or machine learning applications that need access to large samples for model construction or verification. However, that operation is expensive. It is even more costly if the data is transported over a network connection, which is necessary if the database server runs on a separate machine or in the cloud.

Result set serialization has a significant impact on overall system performance. Figure 1 shows the time taken to run the simple SQL query "`SELECT * FROM lineitem`" on various data management systems. We can see large differences between systems and disappointing performance overall. Modern data management systems need a significant amount of time to transfer a small amount of data from the server to the client, even when they are located on the same machine.

Because of the large cost of data export, many analysts settle for exporting small samples from the database. This way, data export is not a bottleneck in their analysis pipelines. However, this reduces the accuracy of their machine learning and classification algorithms.

The issue of slow result export has been identified before. A large amount of previous work focuses on avoiding data export by performing the computations in the database instead of exporting the data. However, these solutions require large, system-specific, overhauls of existing pipelines and are difficult to implement and optimize. There is little to no scientific work done on improving result set serialization.

In this paper, we perform a rigorous examination on the design of existing client protocols. We analyse how they perform when transferring various data sets in different network scenarios, and examine why they show this performance. We explore the design space of client protocols with experiments, and look at the different trade-offs that are made when designing a client protocol.

The main contributions of this paper are:

- We benchmark the result set serialization methods used by major database systems, and measure how they perform when transferring large amounts of data in different network environments. We explain how these methods perform result set serialization, and discuss the deficiencies of their designs that make them inefficient for transfer of large amounts of data.

- We explore the design space of result set serialization and investigate numerous techniques that can be used to create an efficient serialization method. We extensively benchmark these techniques and discuss their advantages and disadvantages.

- We propose a new column-based serialization method that is suitable for exporting large result sets. We implement our method in the Open-Source database system MonetDB, and demonstrate that it performs an order of magnitude better than the state of the art. The implementation of our method is available as Open Source software.