Preface

In the field of disk-based parallel database management systems exists a great variety of solutions based on a shared-storage or a shared-nothing architecture. In contrast, main memory-based parallel database management systems are dominated solely by the shared-nothing approach as it preserves the in-memory performance advantage by processing data locally on each server. We argue that this unilateral development is going to cease due to the combination of the following three trends: (a) Nowadays, network technology features remote direct memory access (RDMA) and narrows the performance gap between accessing main memory inside a server and of a remote server to and even below a single order of magnitude. (b) Modern storage systems scale gracefully, are elastic, and provide high-availability. (c) A modern storage system such as Stanford's RAMCloud even keeps all data resident in main memory. Exploiting these characteristics in the context of a main-memory parallel database management system is desirable. The advent of RDMA-enabled network technology makes the creation of a parallel main memory DBMS based on a shared-storage approach feasible.

This work describes building a columnar database on shared main memory-based storage. It discusses the resulting architecture (Part I), the implications on query processing (Part II), and presents an evaluation of the resulting solution in terms of performance, high-availability, and elasticity (Part III).

We use Stanford's RAMCloud as shared-storage, and the self-designed and developed in-memory AnalyticsDB as relational query processor on top. AnalyticsDB encapsulates data access and operator execution via an interface which allows seamless switching between local and remote main memory, while RAMCloud provides not only storage capacity, but also processing power. Combining both aspects allows pushing-down the execution of database operators into the storage system. We describe how the columnar data processed by AnalyticsDB is mapped to RAMCloud's key-value data model and how the advantages of columnar data storage can be preserved.

The combination of fast network technology and the possibility to execute database operators in the storage system opens the discussion for site selection. We construct a system model that allows the estimation of operator execution costs in

terms of network transfer, data processed in memory, and wall time. This can be used for database operators that work on one relation at a time—such as a scan or materialize operation—to discuss the site selection problem (data pull vs. operator push). Since a database query translates to the execution of several database operators, it is possible that the optimal site selection varies per operator. For the execution of a database operator that works on two (or more) relations at a time, such as a join, the system model is enriched by additional factors such as the chosen algorithm (e.g. Grace- vs. Distributed Block Nested Loop Join vs. Cyclo-Join), the data partitioning of the respective relations, and their overlapping as well as the allowed resource allocation.

We present an evaluation on a cluster with 60 nodes where all nodes are connected via RDMA-enabled network equipment. We show that query processing performance is about 2.4x slower if everything is done via the data pull operator execution strategy (i.e. RAMCloud is being used only for data access) and about 27 % slower if operator execution is also supported inside RAMCloud (in comparison to operating only on main memory inside a server without any network communication at all). The fast-crash recovery feature of RAMCloud can be leveraged to provide high-availability, e.g. a server crash during query execution only delays the query response for about one second. Our solution is elastic in a way that it can adapt to changing workloads (a) within seconds, (b) without interruption of the ongoing query processing, and (c) without manual intervention.

Palo Alto, CA, USA Christian Tinnefeld
April 2015

http://www.springer.com/978-3-319-20710-0

Building a Columnar Database on RAMCloud
Database Design for the Low-Latency Enabled Data Center
Tinnefeld, C.
2016, XIX, 130 p. 37 illus., Hardcover
ISBN: 978-3-319-20710-0