Diss. ETH No. 17144

REPLICATION IN A DATABASE CLUSTER WITH FRESHNESS AND CORRECTNESS GUARANTEES

A dissertation submitted to the

SWISS FEDERAL INSTITUTE OF TECHNOLOGY ZURICH

for the degree of

DOCTOR OF SCIENCES

presented by

Fuat Akal

Master of Science in Computer Science,
Hacettepe University, Ankara, Turkey

born 14.07.1974
citizen of Turkey

accepted on the recommendation of

Prof. Dr. Moira Norrie , examiner
Prof. Dr. Hans-Jörg Schek, co-examiner
Prof. Dr. Yuri Breitbart , co-examiner

2007

Zusammenfassung

Diese Dissertation beschreibt eine Middleware zur Koordination von Datenbank-Clustern welche eine skalierbare Infrastruktur zur kombinierte Bearbeitung von Online-Transaction-Processing (OLTP) Anfragen sowie auch Online-Analytical-Processing (OLAP) Anfragen bietet. Der verfolgte Ansatz zum Aufbau eines derartigen Datenbankclusters basiert auf Standard Hard- und Software, wie z.B. herkömmliche Datenbanksysteme und gewöhnliche Arbeitsplatzrechner / Serverhardware. Die beschriebene Middleware bildet eine Zwischenschicht oberhalb solcher Komponenten, die es erlaubt, Änderungen welche an den OLTP-Knoten auftreten, in identischer Reihenfolge der Commits an den OLAP-Knoten zu wiederholen. Ein wichtiger Aspekt der vorgeschlagenen Middleware stellt die gleichzeitige garantierte "Frische" der Daten und Konsistenzwahrung dar, welche keine Verminderung der Performanz zur Folge hat.

In dieser Arbeit werden im ersten Abschnitt zunächst die grundlegenden Hintergrundinformationen zu Cluster-Computing, Hardware-Architekturen, physischem Datenbankentwurfsalternativen, paralleler Anfragebearbeitung, Cluster-Organisationen und Replikationsverfahren dargestellt.

Hierauf folgt im zweiten Abschnitt eine Vertiefung in Anfrage-Routing in Datenbankclustern. Einen wichtigen Beitrag dieser Arbeit stellt dabei die Erörterung eines hybriden physischen Datenbankschemas über die Knoten des Clusters hinweg dar, welches die Anfrageabarbeitung unterstützt. Im Weiteren folgt die Beschreibung einer darauf aufbauenden Weiterentwicklung des erwähnten Routingalgorithmus, um derartige Schemaentwürfe und -verteilung der Daten geschickt für die Routingentscheidung ausnutzen zu können.

Im dritten Abschnitt werden Änderungen des Datenbestands in die Betrachtung miteingeschlossen und das PDBREP-Protokoll beschrieben, ein fortschrittliches Replikationsprotokoll für Datenbankcluster. Das PDBREP-Protokoll besitzt herausragende Konsistenz- wie auch Geschwindigkeitseigenschaften sowohl für synchrone als auch asynchrone Replikationsausführung. Obwohl dieses Protokoll intern eine asynchrone Replikation nutzt, erscheint es für den Anwender wie eine synchrone Ausführung. Es wurde entworfen, um eine schnellere Ausführung von komplexen analytischen Anfragen bei hohen Änderungsraten zu erreichen. Darüber hinaus erlaubt es dem Anwender als Quality of Service-Parameter die "Freshness"-Anforderungen einzelner Anfragen festzulegen. Zusätzlich unterstützt es auch beliebige physische Datenlayouts welche eine gezielte Anpassung einzelner Clusterknoten für die Nutzung für bestimmte Anfragetypen ermöglicht.

Im vierten Abschnitt wird das PDBREP-Protokoll für die Anwendung in anderen Feldern, hier Data Grids, generalisiert. Es werden die Vorraussetzungen und Einschränkungen bestehender Replikationsansätze für Data Grids beschrieben. Hiernach wird das PDBREP-Protokoll weiter ausgebaut, um es für die Verwendung in Data Grids anzupassen. Es werden dabei jene Annahmen herausgearbeitet, welche für den Entwurf eines Replikationsprotokolls für Data Grids aufgegeben und welche zusätzlich berücksichtigt werden sollen. Daran anschließend wird eine Architektur für die Replikation in Data Grids vorgeschlagen.

Abgeschlossen wird diese Arbeit durch eine umfassende experimentelle Untersuchung, welche die theoretischen dargelegten Vorteile in der praktischen Anwendung

bestätigt. Das vorgeschlagene Replikationsprotokoll und der Routingalgorithmus wurden in einem Prototypen einer Koordinationsmiddleware implementiert. Mehrere Experimente wurden auf Grundlage der im TPC-R Benchmark definerten Datenbanken, Anfragen und Änderungen durchgeführt. Die Experimente zeigen, dass das PDBREP-Protokoll eine gute Leistung selbst bei hohen Änderungsraten und "Freshness"-Anforderungen erreicht. Die Experimente zeigen ebenfalls, dass das PDBREP-Protokoll unabhängig von den darunterliegenden Datenbanksystemen ist und ein gleichartiges Verhalten für unterschiedlichen Datenbanken erreicht. Dies erübrigt jedoch nicht weitere Detailanpassungen des Protokolls für den Fall, dass spezielle Funktionalität der Datenbankkomponente wie beispielweise Snapshot Isolation ausgenutzt werden soll. Bedingt durch die Grösse des Clusters zeigen die Experimente eine geringfügige Verbesserung wenn ein gemischtes physisches Datenbankschema verteilt über die Knoten des Clusters eingesetzt wird. Im besonderem wird das Potential zur Verbesserung des Gesamtdurchsatzes eines derartigen Entwurfs und einem Routingalgorithmus basierend auf dem Datenbankentwurf aufgezeigt.

Abstract

This thesis presents a coordination middleware for database clusters which provides a scalable infrastructure for combined online transaction processing (OLTP) and online analytical processing (OLAP) workloads. The pursued approach for building such database clusters is based on standard hardware and software components, i.e., off-the-shelf PCs and databases. The presented middleware provides a layer on top of such components, which provides replaying of updates that occur at the OLTP node(s) on OLAP nodes while preserving the commit order of the updates. An important aspect of the proposed middleware is that it ensures freshness and consistency guarantees at the same time without causing any performance degradation.

In the beginning, we present a foundational background information regarding cluster computing, hardware architectures, physical database design alternatives, query parallelism, cluster organization and replication schemes.

As a second step, we focus on query routing in database clusters. An important contribution of this work appears at this step where we discuss a use of the mixed physical design schemes across the cluster to facilitate query executions. In order to exploit such design schemes, we developed a routing algorithm which takes into account the design of the data when making routing decisions.

As a third step, we also take into account the updates and present a sophisticated replication protocol PDBREP for database clusters. The PDBREP protocol provides consistency and performance characteristics of synchronous and asynchronous replication schemes respectively. Although it is internally asynchronous, it looks synchronous from user's perspective. It is designed to provide faster evaluations of complex analysis queries under higher update rates. In addition, it allows users to specify their freshness demands attached to the queries as a quality of service parameter. Furthermore, it also supports arbitrary physical data layouts which allow customization of cluster nodes for utilization of certain types of queries.

As a fourth step, we give a generalization of the PDBREP protocol for more general settings, i.e. data grids. We discuss requirements and shortcomings of the existing replication approaches for data grids. We further elaborate on the PDBREP protocol in order to adopt it for data grids. We identify its assumptions that should be dropped and further considerations that should be taken into account while designing a replication protocol for data grids. Then, we propose a replication architecture for data grids.

Finally, we give an extensive experimental evaluation of this work. We implemented the proposed replication protocol and routing algorithms in a prototype of a coordination middleware. We conducted several experiments by using the database, queries and updates defined in the TPC-R benchmark. The experiments show that the PDBREP protocol delivers a good performance even with high update rates and freshness requirements. The experiments also show that the PDBREP protocol is independent of the underlying component database system and behaves similarly for different databases. Nevertheless, this does not avoid us to do customizations in the protocol when desired to exploit a special functionality of the component database like snapshot isolation. Due to the size of the cluster we used for the experiments, deploying a mixed physical design scheme across the cluster shows a bit of improvement. However, we show the potential of such

a design and a routing algorithm based on the data design in order to improve overall throughput.