Database Sharding:
To Provide Fault Tolerance and Scalability of Big Data on the Cloud

Sikha Bagui, Department of Computer Science, University of West Florida, Pensacola, FL, USA
Loi Tang Nguyen, Naval Education and Training, Development and Technology Center (NETPDT), Pensacola, FL, USA

ABSTRACT
In this paper, the authors present an architecture and implementation of a distributed database system using sharding to provide high availability, fault-tolerance, and scalability of large databases in the cloud. Sharding, or horizontal partitioning, is used to disperse the data among the data nodes located on commodity servers for effective management of big data on the cloud.

Keywords: ACID (Atomicity, Consistency, Isolation, Durability), Database Clustering, Database Sharding, Horizontal Data Partitioning

1. INTRODUCTION
As the digital revolution heralds into a new era with Big Data on the cloud, the idea of a centralized data storage has to be modified to accommodate availability, scalability, reliability and manageability. Higher performance and lower cost become a challenge. This can be accommodated by database sharding.

The architecture of a sharded database is comprised of multiple computed nodes deployed across commodity severs to provide continuous uptime operation in the event of hardware or network failure. By leveraging on database sharding or horizontal partitioning, the database is divided into smaller chunks or shards across multiple data nodes in the cluster. Each node contains and is responsible for its own subset of the data to create a shared-nothing environment.

Unlike the shared-disk clustered database, where data can be accessed from all cluster nodes and thus can cause contention during simultaneous reads and writes, shared-nothing clustered database’s nodes operate on their own subset of data. Data nodes are replicated to provide redundancy, and thus provide high availability and scalability. In a worst-case scenario, if a node, containing a subset data of a table, and the replicated node become unavailable, the other nodes, with a different subset of data, will remain online and available.

DOI: 10.4018/IJCAC.2015040103
The business application continues to perform transactions and access data from the remaining available data nodes. These database operations will execute successfully and transparently to the application if the application does not access the unavailable data node. The failed node can be examined, recovered, and rejoin the cluster when it is ready, without affecting any other nodes.

Due to economies of scale, the incurred cost of operating the database needs to be substantially reduced. Database sharding alleviates cost in two ways (1) cost of operation and maintenance - Total Cost of Ownership (TCO) and (2) cost of accessing and processing data. Instead of a centralized storage-area network (SAN), which introduces a single point of failure, database clustering addresses cost (1) by providing a shared-nothing environment that enables the database to scale horizontally on low cost commodity hardware. By wisely partitioning data rows horizontally in a cluster using range or hash partitioning, subsets of data can be accessed separately on each individual data node, which significantly influences the performance capacity while reducing the cost of accessing and processing the data.

Database sharding solves the issue of manageability by partitioning tables and indexes into more manageable units. Database administrators can use database sharding’s Separation of Concerns design concept to pursue the management of data. Since each node is a representation of a portion of the data, maintaining the entire data table can be accomplished in succession for each individual node without affecting any other data nodes. For example, a typical usage for a database administrator is to archive or back up outdated data to save on storage. Rather than archiving the entire data table, a database administrator could archive a single data node containing the data for a particular partition, i.e. year. The archived data can be compressed and transferred to a less-expensive storage tier at a lower TCO cost.

For many years, the perception of gaining more database throughput was by building bigger and better database servers with supplementary multi-cores CPU, faster disk drives, and higher bandwidth and low latency memory. However, according to the law of diminishing returns, all three factors, CPU, disk drive, and memory, have to be added proportionally to provide maximum performance. If only one of the three factors is increased while the others remain constant, then the system will yield lower marginal returns. As business applications grow and are stored on the cloud, databases need to provide high performance to handle the increasing workloads. Not only does database sharding alleviate the TCO cost, but it is also a factor for gaining high performance. Since sharding divides the table into smaller partitions and distributes it across multiple data nodes, the number of records in each node is smaller than whole table. This technique improves searching performance by reducing the index size and limiting the number of records to be traversed.

In this paper, we use MySQL Database Cluster to demonstrate and discover the capabilities of database sharding. We will provide the implementation details to build a sharded database system. After the implementation section, we present some examples of databases that we sharded using our implementation.

The rest of the paper is organized as follows. In section 2 we discuss related works. The implementation of sharded database architecture is presented in Section 3. Section 4 presents experimental results and Section 5 presents the discussion and conclusions.

2. RELATED WORKS

Traditional non-sharded databases are usually deployed on a single database server. If the server fails, the entire business applications crumbles and the system becomes unavailable. The concept of improving database reliability, high availability, maintainability, high performance, and dynamically adapting to increasing workloads has been around for decades. Several works have been done on database clustering and database sharding to address the above issues. In 1999, Pattishall (Pat06) proposed his
Related Content

Do You Need Content Management System?
www.igi-global.com/chapter/you-need-content-management-system/43972?camid=4v1a

Using System Dynamics to Analyze Customer Experience Design
www.igi-global.com/article/using-system-dynamics-analyze-customer/45931?camid=4v1a

How IT-Enabled Services Can Help to Change our World Building Networks for the Energy-Efficient City of Tomorrow
www.igi-global.com/article/enabled-services-can-help-change/60734?camid=4v1a
Does the Internet Increase Fundraising Revenues of Nonprofit Organizations?: An Economic Analysis
www.igi-global.com/chapter/does-internet-increase-fundraising-revenues/50242?camid=4v1a