CISC 525-90-2016 - Assignment 5
Sharding

Nelson Corrocher

July 2016

1 Definition

Sharding, also called **Horizontal Partitioning**, is a method of splitting and storing a single logical dataset in multiple databases. By distributing the data among multiple machines, a cluster of database systems can store larger dataset and handle additional requests. Sharding is necessary if a dataset is too large to be stored in a single database. Moreover, many sharding strategies allow additional machines to be added. Sharding allows a database cluster to scale along with its data and traffic growth.

There exist various strategies to distribute data into multiple databases and each database usually takes different approaches, having pros and cons depending on database assumptions. For example, some operations may need to search through many databases to find the requested data. These are called **cross-partition operations** and they tend to be inefficient. **Hotspots** are another common problemhaving uneven distribution of data and operations. Hotspots largely counteract the benefits of sharding.

For this reason, it's crucial to understand such assumptions and limitations when designing a solution.

2 Sharding in HBase

HBase implements an Auto-Sharding feature, which simply means that tables are dynamically distributed by the system when they become too large. The basic unit of scalability, that provides the horizontal scalability, in HBase is called Region. Regions are a subset of the tables data, and they are essentially a contiguous, sorted range of rows that are stored together. Initially, there is only one region for a table. When regions become too large after adding more rows, the region is split into two at the middle key, creating two roughly equal halves. The main components of HBase are:

- **HMaster**: coordinates the HBase Cluster and is responsible for administrative operations, like assigning regions to servers or keeping META table updated.
- **RegionServer**: can serve one or more Regions. Each Region is assigned to a Region Server on startup and the master can decide to move a Region from one Region Server to another as the result of a load balance operation. The Master also handles Region Server failures by assigning the region to another Region Server.
- **Zookeeper:** this is not a component of HBase but a part of Hadoop Ecosystem. It is a process used for interprocess communications between elements in the distributed system. HBase uses Zookeeper as the way the HMaster and RegionServers communicate with each other.
- **META table:** stored by HMaster in Zookeeper (external locator service); by reading META, one can identify which region is responsible for one key. This means that for read and write operations, the master is not involved at all, and clients can go directly to the Region Server responsible to serve the requested data.

To read and write data, clients need to consult the META table (not HMaster) and access the data directly from RegionServers.

3 Pros and Cons

As the table starts to perform poorer due to size, the auto-sharding feature distribute regions to different servers. No only it increases the parallel processing speed time, since now the same table is now being processed by two hardware units, but sometimes it is possible to avoid accessing a region that doesn't contain the information needed, reducing the workload on the system as a whole.

Regarding availability, since HBase store its database in HDFS files, its availability shares the same benefits and constraints that hadoop system provides.

There are some pitfalls in this approach, however:

- A logical shard (data sharing the same partition key) must fit in a single node. This is the most important assumption, and is the hardest to change in future. A logical shard is an atomic unit of storage and cannot span across multiple nodes. In such a situation, the database cluster is effectively out of space. Having finer partitions mitigates

this problem, but it adds complexity to both database and application. The cluster needs to manage additional partitions and the application may issue additional cross-partition operations.

- In the primary keys used for region sharding are not uniformly distributed, the regions created will have unbalanced workloads and data allocations, creating hotspots (as mentioned above).
- In a range-partitioned sharding scheme, inserting data in partition key order creates hot spots. Only the last range will receive inserts. This partition range will split as it becomes large. However, out of the split ranges, only the latest range will receive additional writes. The write throughput of a cluster is effectively reduced to a single node.

References

[1] How Sharding Works. (2014). Medium. Retrieved 30 July 2016, from https://medium.com/@jeeyoungk/how-sharding-works-b4dec46b3f6#.7mlw0750o

[2] HBase - Who needs a Master? : Apache HBase. (2016). blogs.apache.org. Retrieved 30 July 2016, from https://blogs.apache.org/hbase/entry/hbase_who_needs_a_master

Total words (excluding title and references): 728 words