Quiz on Bigtable

Test your understanding of concepts related to the design of the Bigtable system.

Question 6

Does Bigtable provide strong consistency? Why or why not?

Hide Answer

Bigtable provides strong consistency at the level of single-row reading and writing operations. Row changes in Bigtable are atomic. However, if operations involve multiple rows from the same or multiple tablets, Bigtable does not provide any guarantees, and the application needs to take care of such scenarios (probably using mechanisms such as two-phase locking).

The two main building blocks of Bigtabe (Chubby and GFS) use synchronous replication and it facilitate Bigtable to provide its row-level atomic guarantees.

Question 1

What were the primary motives for inventing Bigtable?

Hide Answer

The primary motives for Bigtable were:

- The ability to store a large amount of data in any column, and the table's ability to store enormous data overall (as compared to a relational database).
- Supporting applications that need very high write and read throughput.
- Supporting sparse tables (where for a row, all the columns might not be present).

Question 2

What are Bigtable column families, and what are their benefits?

Hide Answer

The Column keys are organized into sets called column families. These column families constitute the primary unit of access control. Normally, columns that are related to one another are combined into a column family for fast reading and writing.

Bigtable tries to keep the columns of a column family nearby because they are likely to be used together in queries. Such nearness enables faster reads/writes compared to not using nearby optimizations. Additionally, the constraints on the number of column families/columns in a family usually arise from the fact that there are so many things that can be put nearby.

Question 3

What happens when a Bigtable tablet server fails?

Hide Answer

The Bigtable manager detects such failures and distributes the tablets, which were previously the responsibility of the failed tablet server, to other healthy servers.

Further details:

The Bigtable manager is in charge of determining whether a tablet server is no longer servicing its tablets and reallocating them as quickly as feasible. The manager regularly requests each tablet server for the condition of its lock on the file in Chubby to determine when a tablet server is no longer servicing its tablets.

If a tablet server indicates that it has failed its lock, or if the manager has not been able to access a server in the past few tries, the manager attempts to obtain an exclusive lock on the server's file in Chubby service.

If the manager is successful in obtaining the lock, this indicates that Chubby is still functioning and the tablet server is either inactive or experiencing difficulty connecting to Chubby. In this case, the manager deletes the server file of the tablet server to make sure that it doesn't serve anymore. After deleting a server's file, the manager shifts all previously assigned tablets of the faulty server into the group of unassigned tablets. Later, these tablets will be assigned to some healthy servers.

Question 4

What happens when a Bigtable manager server fails?

Hide Answer

When Google's Cluster Management System discovers that there is no current manager, it initiates the creation of one. Before functioning as a Bigtable manager, the new manager must first get the lock on a Chubby file to ensure there is only one manager at all times. At start-up, the manager performs the following processes.

1. To avoid repeated manager instantiations, the manager acquires a distinct manager lock in Chubby.
2. The manager searches Chubby's "servers" directory for active tablet servers.
3. The manager checks which tablets are associated with each active tablet server by interacting with it.
4. To understand the whole collection of tablets, the manager analyses the Metadata table. When this search detects a tablet that has not yet been assigned, the manager inserts it into the list of unassigned tablets. Likewise, the manager creates a pool of unassigned tablet servers that are eligible for tablet assignment. This information is used by the manager to allocate unassigned tablets to relevant tablet servers.

The manager is in charge of all metadata activities (for example, table and column family creations) and other management functions such as monitoring and load-balancing tablet servers. As a result, if the manager fails, the clients will be unable to make schema modifications, but they can still access data during this period as data communication is directly between clients and tablet servers. Clients can do normal writing operations as well (as long as schema changes are not required).

Question 5

How does Bigtable ensure system reliability?

Hide Answer

Bigtable depends on two services Chubby and GFS. Each of these systems uses a replication method to increase the reliability of the system.

A Chubby system, for example, is often composed of five replicas, with one serving as the primary and the rest four serving as secondary replicas. If the primary fails, one of the replicas is chosen to be the leader, reducing Chubby's downtime.

Similarly, GFS replicates data over several chunkservers (usually three).

Tablet server failures are also managed by the Bigtable manager server. The manager is responsible for determining whether a tablet server is no longer serving its tablets and reassigning them as quickly as possible. The manager regularly requests each tablet server for the condition of its lock on the file in Chubby to determine when a tablet server is no longer servicing its tablets. Clients don't interact with this manager for reading and writing (they go directly to the tablet servers). That way, Bigtable's manager usually has a light load.

Collectively Bigtable's reliance on dependable building blocks (Chubby, GFS), along with its own machinery, helps to keep the system going.

Question 6

Does Bigtable provide strong consistency? Why or why not?

Hide Answer

Bigtable provides strong consistency at the level of single-row reading and writing operations. Row changes in Bigtable are atomic. However, if operations involve multiple rows from the same or multiple tablets, Bigtable does not provide any guarantees, and the application needs to take care of such scenarios (probably using mechanisms such as two-phase locking).

The two main building blocks of Bigtabe (Chubby and GFS) use synchronous replication and it facilitate Bigtable to provide its row-level atomic guarantees.