ArchitectureCommitLog

The CommitLog class is a singleton that provides a facade for global CommitLog operations. For 1.1, the CommitLogAllocator is delegated the activity of managing a set of CommitLogSegment instances, each of which corresponds to a file on disk containing a sequence of serialized RowMutation objects.

The CommitLogExecutorService classes manage a single thread that executes the actual CommitLog write tasks and controls fsync activity. A configuration directive "commitlog_sync" is provided to control which executor service class is selected.

By default, "commitlog_sync" is set to periodic, which uses the PeriodicCommitLogExecutorService class as the executor implementation. This class performs the writes as soon as they appear on the execution queue and they are finished immediately. Another thread periodically enqueues a sync operation on the execution queue as specified by the "commitlog_sync_period_in_ms" directive in cassandra.yaml.

If "commitlog_sync" is set to batch, the BatchCommitLogExecutorService class is used as the implementation, which groups multiple mutations over a time period window defined by the "commitlog_sync_batch_window_in_ms" directive and executes them in batches. After each batch write and before completing the tasks and thus acknowledging the write to the client, it performs a sync of the commit log.

The CommitLogSegment class keeps track of which column families have been modified in memory using a hash map called cfLastWrite. cfLastWrite has one entry per ColumnFamily, consisting of an offset, indicating the position in the CommitLog file where the last write took place for a particular ColumnFamily. When a ColumnFamily memtable is flushed, the segments containing those mutations are marked as clean, or, if a mutation was written to the segment since the flush was initiated, the dirty marker is advanced forward to the point of the last unflushed write.

As of 1.1, segment files are pre-allocated up to a fixed size (128MB) and the CommitLogAllocator manages a thread queue that pre-allocates these segments and doles them out. When the commit log segment fills up with mutations, it is rotated out and an empty segment is used. Once all of the mutations in a segment file have been flushed to SSTable files and a segment is no longer needed, it is renamed and "recycled," making it available for reuse. Segments that existed prior to launching Cassandra will be recycled once they have gone through the recovery process. The allocator eagerly allocates empty segments ahead-of-time to instantly provide more capacity if needed, so an empty segment is always available.

Recovery during a restart,

- Each CommitLogSegment is iterated in ascending time order.
- The segment is read from the lowest replay offset among the ReplayPositions read from the SSTable metadata.
- For each log entry read, the log is replayed for a ColumnFamily CF if the position of the log entry is no less than the ReplayPosition for CF in the most recent !SSTable metadata.
- When log replay is done, all Memtables are force flushed to disk and all commitlog segments are recycled.