Durability

Durability is the property that writes, once completed, will survive permanently, even if the server is killed or crashes or loses power. This requires calling fsync to tell the OS to flush its write-behind cache to disk.

The naive way to provide durability is to fsync your data files with each write, but this is prohibitively slow in practice because the disk needs to do random seeks to write the data to the write location on the physical platters. (Remember that each seek costs 5-10ms on rotational media.)

Instead, like other modern systems, Cassandra provides durability by appending writes to a commitlog first. This means that only the commitlog needs to be fsync'd, which, if the commitlog is on its own volume, obviates the need for seeking since the commitlog is append-only. Implementation details are in ArchitectureCommitLog.

Cassandra's default configuration sets the `commitlog_sync` mode to `periodic`, causing the commitlog to be synced every `commitlog_sync_period_in_ms` milliseconds, so you can potentially lose up to that much data if all replicas crash within that window of time. This default behavior is decently performant even when the commitlog shares a disk with data directories. You can also select `batch` mode, where Cassandra will guarantee that it syncs before acknowledging writes. To avoid syncing after every write, Cassandra groups the mutations into batches and syncs every `commitlog_batch_window_in_ms`. When using this mode, we strongly recommend putting your commitlog on a separate, dedicated device, as described above.

Users familiar with PostgreSQL may note that `commitlog_sync_period_in_ms` and `commitlog_batch_window_in_ms` correspond to the PostgreSQL settings of wal_writer_delay and commit_delay, respectively.