**WRITE PATHS IN CASSANDRA :**

**TERMINOLOGY OF APACHE CASSANDRA:**

* Gossips
* Partitioners
* Snitchers
* SSTables
* Tombstones
* Compactions

Cassandra uses murmur3 partitioners and murmur3 algorithms to create Tokens

**CASSANDRA DATA WRITE PATH** :

**HOW DATA IS WRITTEN** :

Data is processed by Cassandra at multiple points along the write path, commencing with the instantaneous logging of a write and concluding with a write to disc

* Logging data in the commit log
* Writing data to the memtable
* Flushing data from the memtable
* Storing data on disk in SSTables

**LOGGING DATA IN COMMIT LOG :**

When a write happens, Cassandra appends writes to the commit log on disc in addition to storing the data in a memory structure known as a memtable for adjustable durability. Every write that is done to a Cassandra node ends up in the commit log; these durable writes don't disappear when a node loses power. Cassandra searches up data partitions by key in a write-back cache called the memtable. The memtable is flushed once it has stored writes in a sorted manner up to a specified maximum.

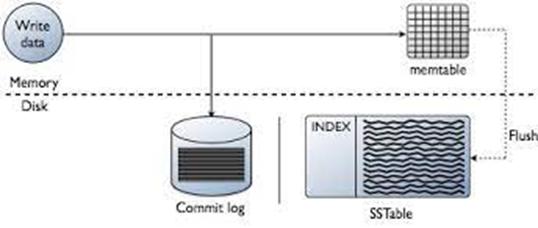
**FLUSHING DATA FROM THE MEMTABLE :**

Cassandra writes the data in the memtable-sorted order to disc in order to flush the data. Additionally, a partition index mapping the tokens to a disc location is established on the disc. The memtable is placed in a queue that is flushed to disc when either the commitlog space or the memtable content reach the configured threshold in megabytes. The memtable\_heap\_space\_in\_mb or memtable\_offheap\_space\_in\_mb setting in the cassandra.yaml file can be used to configure the queue. If there is more data to be flushed than memtable\_cleanup\_threshold . Writes are blocked by Cassandra until the next flush is successful. Nodetool flushor nodetool drain (which flushes memtables without listening for connections to other nodes) can be used to manually flush a table. It is advised as best practise to flush the memtable before restarting the nodes in order to minimise the commit log replay time. Replaying the commit log allows the writes that were present in the memtable prior to the node stopping to be restored in case it ceases functioning.

After matching data in the memtable is flushed to an SSTable on disc, data in the commit log is cleared.

**STORING DATA ON DISC SSTABLES :**

Each table has its own SSTables and Memtables. Tables can share the commit log. After the memtable has been flushed, SSTables cannot be changed. As such, a partition is usually spread across several SSTable files for storage. There are more SSTable structures available to help with read operations

The SSTables are disk-based files. With Cassandra 2.2 and later, the naming standard for SSTable files was modified to shorten the file path. The data directory where the data files are kept differs depending on the installation. Every table is stored in a directory within the data directory for every keyspace. 

Each table that Cassandra builds has its own subfolder, which enables you to symlink it to a desired physical drive or storage volume. This allows for better performance by moving highly active tables to faster media, such SSDs, and it also partitions tables across all attached storage devices for improved I/O balancing at the storage layer.