Cassandra 3.0数据修复机制 - 博客频道

参考
https://docs.datastax.com/en/cassandra/3.0/cassandra/operations/opsRepairNodesTOC.html

前提：
每个数据副本N，写一致性级别是W，读一致性级别是R

Hinted Handoff(提示移交)：写修复

写操作是会发送N个写请求，但是只统计W个。对于另外N-W个节点，如果写失败，则记录hint。

hint内容

- target ID：目标节点
- hint ID：数据时间戳
- message ID：Cassandra版本
- blob：数据

写时修复包括五种情况：

- 一致性级别不满足
  当用户指定的一致性级别不满足，或协调者挂了，则抛出UnavailableException异常。

- 一致性级别Any
  将hint写入协调者节点也当做写成功。

- 失效检测机制已经标记节点挂掉

  当Cassandra配置中hinted handoff打开，丢失的写操作在协调者节点以hint的格式存储一段时间T，hint保存在协调者本地hints目录，每十秒更新一次。当节点恢复之后，将hints中的每个hint写到恢复的节点上。如果节点超过max_hint_window_in_ms（3小时）还没恢复，停止写新的hints。

- 还未标记挂掉

  当节点还没来得及被标记挂掉，当写操作超过write_request_timeout_in_ms（10秒），协调者节点返回一个TimeOutException，写操作失败，存储一个hint。当失效节点过多，协调者统计需要写入的hint数量，如果超过一定值，就拒绝写操作并且抛出OverloadedException。

- 目标节点被移除集群
  删除对应此节点的hints

Read Repair：读修复

对于使用了`DataTieredCompactionStrategy`的table，则设置`read_repair_chance`为0。不进行修复。对其他压缩策略，读修复概率一般设置为20%

- 根据一致性级别R进行修复

  读流程读R个，对读到的R个副本进行比较，并将旧的数据进行修复。如果R=1不进行修复。

- 随机读修复

  直接或后台读取所有N个副本的digest信息，进行比较，并修复旧的数据。

Manual repair：反熵修复

Merkle Tree

对数据进行hash，作为叶节点，之后每两个节点构建一个parent节点，直到根节点。

nodetool repair

- nodetool repair

  将此节点负责的数据按token ranges分段，每一段涉及的所有副本节点将参与修复（将涉及的每一段的所有副本修好）。

- nodetool repair -pr

  将只修复这个节点直接负责的数据段的所有副本

- nodetool repair -inc

  增量修复，修复进程的leader发送修复请求到其他相关节点，其他节点根据sstable的元数据中的repairedAt字段判断是否修复过，只对未修复的sstable生成merkle tree。然后汇总到leader，leader比较并进行恢复。恢复完发送一个anticompaction命令，将修复过的和没修复过的range存储到不同的sstable中。修复过的sstable用repairedAt字段标识，值为修复的时间。
  修复和compaction是不互斥的，所以可能还没修复完就被删除了，一会会再次修复。这影响效率，但不影响正确性。
  anticompaction有两种策略：Size-Tiered和leveled
  当一个sstable被修复的range全覆盖，则不进行anticompaction。只更新repairedAt字段。