Elasticsearch 存储方式和管理优化细节

Elasticsearch 的数据存储方式：
Lucene 把每次生成的倒排索引，叫做一个段(segment).然后另外使用一个 commit 文件记录索引内所有的 segment，生成 segment 的数据来源，refresh 到内存中的 buffer。
从写入refresh到文件缓存buffer中默认设置为 1 秒。

Elasticsearch 在把数据写入到内存 buffer 的同时，其实还另外记录了一个 translog 日志。通过translog 日志真正把 segment 刷到磁盘，同时commit 文件进行更新，然后translog 文件才清空。这

一步，叫做 flush。默认设置为：每 30 分钟主动进行一次 flush。

上述两个过程保证数据实时查询和持久化数据。

注：5.0 中还提供了一个新的请求参数：?refresh=wait_for，可以在写入数据后不强制刷新但一直等到刷新才返回。对于日志记录，可以等到时间缓冲后再刷新，不需要保证实时，"refresh_interval":

"10s"；对于归档的数据导入时，可以先设置"refresh_interval": "-1"关闭刷新，导入完后手动刷新即可。

注：为了减小系统开销，小的segment归并成大的segment再提交保存。segment 归并的过程，需要先读取 segment，归并计算，再写一遍 segment，最后还要保证刷到磁盘。5.0后引入Lucene的CMS自动调

整机制，默认设置是 10240 MB；封装了"indices.store.throttle.max_bytes_per_sec" 该配置，不需要再设置。归并线程保持默认即可。index.merge.scheduler.max_thread_count=3

归并策略优化：

index.merge.policy.floor_segment 默认 2MB，小于这个大小的 segment，优先被归并。
index.merge.policy.max_merge_at_once 默认一次最多归并 10 个 segment
index.merge.policy.max_merge_at_once_explicit 默认 forcemerge 时一次最多归并 30 个 segment。
index.merge.policy.max_merged_segment 默认 5 GB，大于这个大小的 segment，不用参与归并。forcemerge 除外。
其实我们也可以从另一个角度考虑如何减少 segment 归并的消耗以及提高响应的办法：加大 flush 间隔，尽量让每次新生成的 segment 本身大小就比较大。

路由：
shard = hash(routing) % number_of_primary_shards routing参数： _id

集群管理：
1.节点下线
Elasticsearch 集群就会自动把这个 IP 上的所有分片，都自动转移到其他节点上。等到转移完成，这个空节点就可以毫无影响的下线了。

```
curl -XPUT 127.0.0.1:9200/_cluster/settings -d '{
  "transient" :{
      "cluster.routing.allocation.exclude._ip" : "10.0.0.1"
   }
}'
```

2.迁移分片
例如负载过高，使用reroute接口下的指令。常用的一般是 allocate 和 move
从 5.0 版本开始，Elasticsearch 新增了一个 allocation explain 接口，专门用来解释指定分片的具体失败理由

3.冷热数据的读写分离
N 台机器做热数据的存储, 上面只放当天的数据。这 N 台热数据节点上面的 elasticsearc.yml 中配置 node.tag: hot；之前的数据放在另外的 M 台机器上。这 M 台冷数据节点中配置 node.tag:

stale

模板中控制对新建索引添加 hot 标签：

```
{
 "order" : 0,
 "template" : "*",
 "settings" : {
   "index.routing.allocation.require.tag" : "hot"
 }
}
```

每天计划任务更新索引的配置，将 tag 更改为 stale，索引会自动迁移到 M 台冷数据节点

```
# curl -XPUT http://127.0.0.1:9200/indexname/_settings -d'
{
"index": {
  "routing": {
    "allocation": {
      "require": {
        "tag": "stale"
      }
    }
  }
}
}'
```

这样，写操作集中在 N 台热数据节点上，大范围的读操作集中在 M 台冷数据节点上。避免了堵塞影响。

4. 检测参数优化

discovery.zen.minimum_master_nodes: 3

discovery.zen.ping_timeout: 100s　//参数仅在加入或者选举 master 主节点的时候才起作用

discovery.zen.fd.ping_timeout: 100s // fd故障检测参数则在稳定运行的集群中，master 检测所有节点，以及节点检测 master 是否畅通时长期有用。

discovery.zen.fd.ping_interval: 10s //间隔

discovery.zen.fd.ping_retries: 10　//重试次数

discovery.zen.ping.unicast.hosts:
["10.19.0.97","10.19.0.98","10.19.0.99","10.19.0.100"] //unicast 单播

5.磁盘限额:
cluster.routing.allocation.disk.watermark.low (默认 85%)
cluster.routing.allocation.disk.watermark.high (默认 90%)

每个节点的分片数，需要把故障迁移问题考虑进去
"routing.allocation.total_shards_per_node" : "5" //需要在创建时考虑

6.在线缩容
从 5.0 版本开始，Elasticsearch 新提供了 shrink 接口，可以成倍数的合并分片数。

7.过滤器
Ingest 节点是 Elasticsearch 5.0 新增的节点类型和功能。节点接收到数据之后，根据请求参数中指定的管道流 id，找到对应的已注册管道流，对数据进行处理，然后将处理过后的数据，按照

Elasticsearch 标准的 indexing 流程继续运行。

8.通过监控任务，check集群健康
curl -XGET 'localhost:9200/_cat/tasks?v'

9.支持chef、ansible、puppet自动化部署

【原创】原创文章，更多关注敬请关注微信公众号。